VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

KLASIFIKACE VZTAHŮ MEZI POJMENOVANÝMI ENTITAMI V TEXTU

CLASSIFICATION OF RELATIONS BETWEEN NAMED ENTITIES IN TEXT

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **Bc. KAREL ONDŘEJ**
AUTHOR

VEDOUCÍ PRÁCE **Doc. RNDr. PAVEL SMRŽ, Ph.D.**
SUPERVISOR

BRNO 2020

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2019/2020

Zadání diplomové práce

Student: **Ondřej Karel, Bc.**
Program: Informační technologie Obor: Inteligentní systémy
Název: **Klasifikace vztahů mezi pojmenovanými entitami v textu**
Classification of Relations between Named Entities in Text
Kategorie: Umělá inteligence
Zadání:

1. Seznamte se s metodami extrakce pojmenovaných entit a vztahů z textu na základě strojového učení.
2. Navrhněte a implementujte systém pro automatickou klasifikaci vztahů mezi entitami, se zaměřením na zvyšování přesnosti pomocí dodatečných informací.
3. Vytvořte systém pro pravidelné rozšiřování extrahovaných dat na základě nově získávaných webových dat.
4. Vyhodnoťte výsledky systému na reprezentativním vzorku dat.
5. Vytvořte stručný plakát prezentující práci, její cíle a výsledky.

Literatura:

- Manning, C. D., Schütze, H., Foundations of Statistical Natural Language Processing, MIT Press, 1999, ISBN 0-262-13360-1.

Při obhajobě semestrální části projektu je požadováno:

- funkční prototyp řešení

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/
Vedoucí práce: **Smrž Pavel, doc. RNDr., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 3. června 2020
Datum schválení: 3. března 2020

Abstrakt

Tato diplomová práce se zabývá extrakcí vztahů mezi pojmenovanými entitami v textu. V teoretické části práce je rozebrána problematika reprezentace přirozeného jazyka pro strojové zpracování. Následně jsou definovány dvě dílčí úlohy extrakce vztahů, a to rozpoznávání pojmenovaných entit a klasifikace vztahu mezi nimi, včetně shrnutí dnešních nejmodernějších řešení. V praktické části práce je navržen systém pro automatickou extrakci vztahů mezi pojmenovanými entitami ze stažených webových stránek. Model pro klasifikaci vztahů mezi entitami je založen na předtrénovaném modelu sítě typu transfomers. V práci jsou porovnány čtyři předtrénované modely, a to BERT, XLNet, RoBERTa a ALBERT.

Abstract

This master thesis deals with the extraction of relationships between named entities in the text. In the theoretical part of the thesis, the issue of natural language representation for machine processing is discussed. Subsequently, two partial tasks of relationship extraction are defined, namely named entities recognition and classification of relationships between them, including a summary of state-of-the-art solutions. In the practical part of the thesis, system for automatic extraction of relationships between named entities from downloaded pages is designed. The classification of relationships between entities is based on the pre-trained transformers. In this thesis, four pre-trained transformers are compared, namely BERT, XLNet, RoBERTa and ALBERT.

Klíčová slova

extrakce vztahů, rozpoznávaní pojmenovaných entit, transformers, BERT, ALBERT, RoBERTa, XLNet, dotrénování

Keywords

relationship extraction, named-entity recognition, transformers, BERT, ALBERT, RoBERTa, XLNet, fine-tuning

Citace

ONDŘEJ, Karel. *Klasifikace vztahů mezi pojmenovanými entitami v textu.* Brno, 2020. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Doc. RNDr. Pavel Smrž, Ph.D.

Klasifikace vztahů mezi pojmenovanými entitami v textu

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana docenta Pavla Smrže. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Karel Ondřej
3. června 2020

Poděkování

Tímto bych chtěl poděkovat svému vedoucímu práce, panu docentu Pavlu Smržovi, za užitečné rady a vedení při vývoji systému a psaní diplomové práce. Dále bych rád poděkoval Elišce Olšanové za jazykovou korekturu textu.

Obsah

Kapitola 1

Úvod

Objem dat na internetu v dnešní době rapidně narůstá a skýtá nám velké množství nových informací. S dostupností velkého množství dat se také rozvíjí oblast strojového zpracování přirozeného jazyka, jejíž součástí je např. získávání informací z textu nebo snaha zjednodušit komunikaci člověka s počítačem. Příkladem úloh z této oblasti může být hledání odpovědí v textu na položené otázky v přirozeném jazyce, strojový překlad do jiného jazyka, nebo extrakce informací a vytváření znalostních bází.

Tato práce se konkrétně zabývá úlohami rozpoznání pojmenovaných entit v textu a klasifikací vztahu mezi nimi. Tradičně se za pojmenované entity považují vlastní jména osob, organizací nebo míst. V průběhu času se přidávaly i další kategorie jako názvy produktů, literárních děl, proteinů, ale i čísla a data. Vztahem jsou myšleny sémantické vazby, ať od obecných vztahů, ve kterých jedna pojmenovaná entita je součástí jiné entity (Báseň *Polednice* je součástí sbírky *Kytice*.), až po velmi konkrétní vztahy, jako interakce mezi dvěma proteiny popsaná ve vědecké publikaci.

Práce volně navazuje na autorovu činnost ve výzkumné skupině Znalostních technologií Fakulty informačních technologií Vysokého učení technického v Brně (KnoT) a bakalářskou práci [31]. Předchozí činnost se zabývala masivním inkrementálním stahováním stránek z internetu. Navržený systém v této práci z inkrementálně stažených webových stránek extrahuje pojmenované entity a klasifikuje vztah mezi nimi do několika desítek kategorií s využitím neuronových sítí. Spojení těchto dvou systémů může být potenciálně využito k „nekonečnému“ vytváření znalostních bází.

První dvě kapitoly se zabývají teoretickým základem celé úlohy. Kapitola 2 popíše obecné modely využívané pro zpracování přirozeného jazyka. Konkrétně si představíme modely pro vytvoření vektorové reprezentace vět, která by dokázala zachytit pro lidi intuitivní syntaktické a sémantické informace. Následně si představíme rekurentní a konvoluční neuronové sítě a sítě typu *transformers*, které dokáží vytvořit kontextovou vektorovou reprezentaci vstupních vět.

Samotným úlohám rozpoznávání pojmenovaných entit a klasifikace vztahu mezi entitami se bude věnovat kapitola 3. Kromě definice úloh a jiných modelů než jsou neuronové sítě si představíme „nejmodernější“ přístupy pro jejich řešení.

O návrhu a implementaci systému pojednává kapitola 4. Kapitola popíše jednotlivé fáze zpracování, od extrakce vět ze vstupních souborů, jejich zpracování, až po uložení extrahovaných vztahů. Model pro klasifikaci je založen na předtrénované síti typu *transformers* a jejím *dotrénování* pro konkrétní úlohu. Systém je vyhodnocen na třech datových sadách. Výsledky experimentů naleznete v kapitole 5.

Kapitola 2

Reprezentace přirozeného jazyka

Společným problémem pro všechny úkoly z oblasti zpracování přirozeného jazyka, včetně rozpoznávání pojmenovaných entit a extrakce vztahů mezi nimi, je vhodná reprezentace přirozeného jazyka pro strojové zpracování. Pro člověka je naprosto přirozené, že dokáže rozpoznat lexikální nebo významovou podobnost mezi dvěma slovy (homonyma[1], synonyma[2], nebo antonyma[3]). Také je pro nás přirozené rozpoznat sémantické vztahy mezi slovy, například vztah mezi slovy otec-syn a matka-dcera. Při strojovém zpracování nám tyto intuitivní znalosti chybí, proto potřebujeme nalézt vhodný způsob pro reprezentaci jednotlivých slov, abychom byli schopni tyto podobnosti zachytit. Této problematice se věnuje první část kapitoly.

Při běžné komunikaci dokážeme také rozpoznat drobné nuance významu slov podle kontextu, v jakém byly použity. Druhá část kapitoly se bude zabývat kontextovou reprezentací slov ve větě, ale i reprezentací celých vět.

2.1 Reprezentace slov

Pro strojové zpracování přirozeného jazyka je výhodné reprezentovat každé slovo vektorem o konstantní délce. Jako nejjednodušší řešení se jeví sestavit slovník všech slov a ten lexikograficky seřadit. Jednotlivá slova lze následně reprezentovat jako vektor ve formátu *one-hot*, kdy na pozici se shodou slova se seřazeným slovníkem bude hodnota 1, jinak 0. Tento způsob má hned několik nevýhod, jako velkou dimenzi prostoru, všechna slova se nemusí nacházet ve slovníku a samotná údržba slovníku je náročná.

Další možností je zobrazit slovo do prostoru reálných čísel s konstantní dimenzí. Nejjednodušším řešením je sestavit vyhledávací tabulku, kdy se každému slovu ze slovníku přiřadí náhodný vektor s uniformním rozložením pravděpodobnosti. Při každém nově nalezeném slově, které se nenachází ve slovníku, se mu přidělí unikátní náhodný vektor, speciální vektor pro neznámá slova z vyhledávací tabulky, nebo se aktualizuje vyhledávací tabulka.

Reprezentace pomocí náhodně inicializovaných vektorů reálných čísel je již praktičtější, ale stejně jako formát *one-hot* neposkytuje žádnou informaci o sémantice slov. Ukázalo se, že při použití neuronových sítí k natrénování reprezentace vzniknou závislosti mezi vektory reprezentující syntakticky a sémanticky podobná slova. Navíc vzniknou závislosti mezi vektory na základě sémantických vztahů mezi slovy [30]. Důsledkem tohoto zjištění

[1] Homonyma jsou slova s podobnou zvukovou nebo grafickou podobou, ale jiného významu i původu.
[2] Synonyma jsou slova stejného nebo podobného významu.
[3] Antonyma jsou slova opačného významu.

je, že na základě vektorové reprezentace například slov otec a syn můžeme určit přibližnou vektorovou reprezentaci slova, které je ve stejném vztahu ke slovu matka, jelikož by měl platit následující vztah:

$$\text{vec}(\text{„otec“}) - \text{vec}(\text{„syn“}) \approx \text{vec}(\text{„matka“}) - \text{vec}(\text{„dcera“}) \ . \tag{2.1}$$

Dále bude popsáno několik modelů pro vektorovou reprezentaci slov ve větě, které dokážou zachytit dříve popsané podobnosti. Každý z modelů může využívat různé přístupy pro naučení reprezentace, ale po naučení je lze nahradit jednoduchou vyhledávací tabulkou. Pomocí vyhledávací tabulky lze následně překládat slova ze slovníku na příslušnou vektorovou reprezentaci.

2.1.1 Word2Vec

Jedná se o skupinu modelů, které převádí slova ve formátu *one-hot* do vektoru reálných čísel [29]. Modely sdílí společnou myšlenku, že významově podobná slova budou v textu obklopovat některá slova častěji než jiná. Oproti jiným architekturám využívajícím hluboké neuronové sítě se modely skládají pouze z jedné vrstvy. Tato architektura nemusí vytvářet natolik kvalitní řešení jako hluboké neuronové sítě, ale může být efektivněji trénovaná na velkých datových sadách s logaritmickou časovou složitostí. Dále si představíme dva modely, které se liší zvoleným přístupem při trénování reprezentace.

Model *CBOW* (*continuous bag-of-words*) se při učení snaží na základě okolních slov predikovat vektorovou reprezentaci aktuálního slova. Model provede projekci předcházejících a následujících slov do určité maximální vzdálenosti na vektorovou reprezentaci. Predikovaná reprezentace aktuálního slova se potom vypočítá jako průměr těchto vektorů.

Model *continuous skip-gram* se naopak pokouší z projekce aktuálního slova predikovat slova v jeho okolí. Při učení se tedy snažíme maximalizovat správnost klasifikace okolních slov ve větě. Pro kvalitní reprezentaci je potřeba modely trénovat na velkých souborech dat, což je časově náročné, proto se využívají již předtrénované modely.

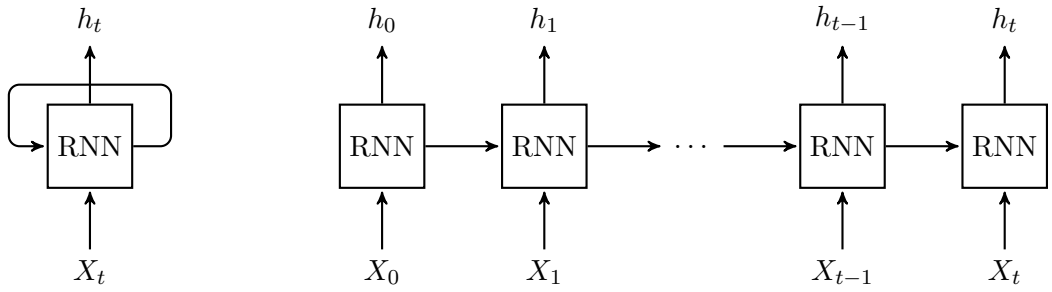
2.1.2 GloVe

Model *GloVe* (*Global Vectors*) je založen na statistické metodě matic společných výskytů [32]. Matice společných výskytů je čtvercová matice o dimenzi počtu slov ve slovníku. Řádky matice nám poskytují ke každému slovu ze slovníku statistiku o počtu výskytů ostatních slov ve stejném kontextu. Kontextem můžeme chápat větu, nebo pevně dané okolí kolem slova. Z matice lze jednoduše určit pravděpodobnost výskytu dvou slov ve stejném kontextu.

Samotné řádky matice nám již tvoří vektorovou reprezentaci slova, jejíž dimenze vektorů je velká a závislá na velikosti slovníku. Pro redukci dimenze vektorů této reprezentace se snažíme sestrojit na základě matice společných záměn zobrazení do vektorového prostoru zachycující podobnost slov s využitím rozdílu dvou různých vektorů. Model lze rychle naučit, je škálovatelný pro velké korpusy a zároveň dosahuje dobrých výsledků pro malé korpusy a malé dimenze vektorů.

2.2 Reprezentace kontextu

Předchozí kapitola pojednávala, jak lze reprezentovat jednotlivá slova pomocí vektoru reálných čísel, aby se zachytil jejich význam. Při použití slova ve větě se může jeho význam



(a) Rekurentní neuronová síť.

(b) Rozvinutá rekurentní neuronová síť v čase.

Obrázek 2.1: Architektura rekurentní neuronové sítě (zkr. RNN). Blok v schématu znázorňuje jednu vrstvu sítě, která je složena alespoň z jednoho neuronu.

mírně, ale i radikálně změnit. V následující kapitole bude představeno několik různých druhů neuronových sítí, které dokážou z vektorové reprezentace významu slov vytvořit reprezentaci kontextu jednotlivých slov nebo celých vět. Velké množství dnešních „nejmodernějších" řešení v oblasti zpracování přirozeného jazyka je založeno právě na těchto modelech, které jsou různě upravovány a kombinovány s dalšími přístupy pro zlepšení výsledků.

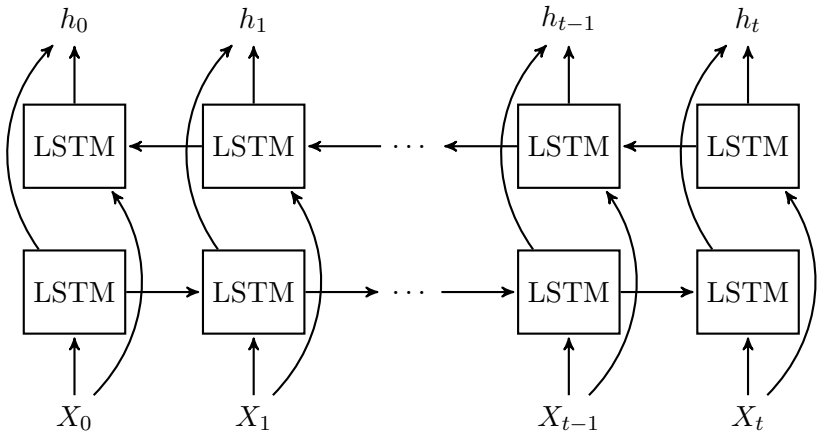
2.2.1 Rekurentní neuronové sítě

Rekurentní neuronové sítě si dokážou oproti běžným neuronovým sítím uchovávat svůj vnitřní stav v průběhu celé klasifikace. To je dosaženo přidáním cyklických vazeb mezi vrstvami, případně zpětné vazby (angl. *self-loop*) vrstvy, nebo samotného neuronu. Na obrázku 2.1a je znázorněna obecná architektura rekurentní neuronové sítě se zpětnou vazbou, kde se vrstva L může podle potřeby lišit. Nejjednodušší rekurentní neuronová síť se může skládat z jedné vrstvy plně propojených neuronů (případně pouze jednoho neuronu), kde každý neuron má svou zpětnou vazbu.

Zpětná vazba je užitečná u sekvenčních úloh, kdy aktuální vstup je závislý na předchozích hodnotách a každá sekvence může mít rozdílnou délku. Obrázek 2.1b demonstruje použití rekurentních sítí pro klasifikaci sekvence o délce t, kde můžeme vidět závislost výstupu na předchozí hodnotě a aktuálním vstupu. Rekurentní neuronové sítě se využívají například pro predikci časových řad, strojový překlad a zpracování přirozeného jazyka.

Učení rekurentních sítí je odlišné od běžných neuronových sítí kvůli nutnosti propagace chyb napříč jednotlivými kroky. Před učením musí být síť rozvinuta v čase, jako na obrázku 2.1b, a následně lze využít algoritmy *backpropagation through time* a *real-time recurrent learning* [39]. S učením rekurentních vah dochází k problému „explodujících" gradientů, kdy nekontrolovatelně rostou, nebo „mizejících" gradientů, kdy se naopak blíží nule.

Rekurentní neuronové sítě se při zpracovávání přirozeného jazyka využívají k zachycení kontextu věty a predikci následujících slov. Pokud k predikci následujícího slova potřebujeme znát slova v jeho těsné blízkosti, vystačíme si se základními rekurentními sítěmi. Například z věty „Na jasné obloze nebylo ani. . . " je jasné, jak bude dále pokračovat, jelikož všechny potřebné informace se nacházejí v těsné blízkosti. Problém nastává při *vzdálených závislostech* (*long-term dependencies*), kdy ke správné predikci musíme znát vzdálený kontext. Určit, jak bude pokračovat věta „Vypadá to, že bude. . . " není úplně jasné bez znalosti kontextu „Na obloze byly bouřkové mraky.", který nemusí nutně přímo předcházet. I když by podle teorie měly být rekurentní sítě schopny tyto závislosti zachytit, v praxi selhávají.



Obrázek 2.2: Architektura obousměrné rekurentní sítě *Bi-LSTM*.

Síť typu LSTM

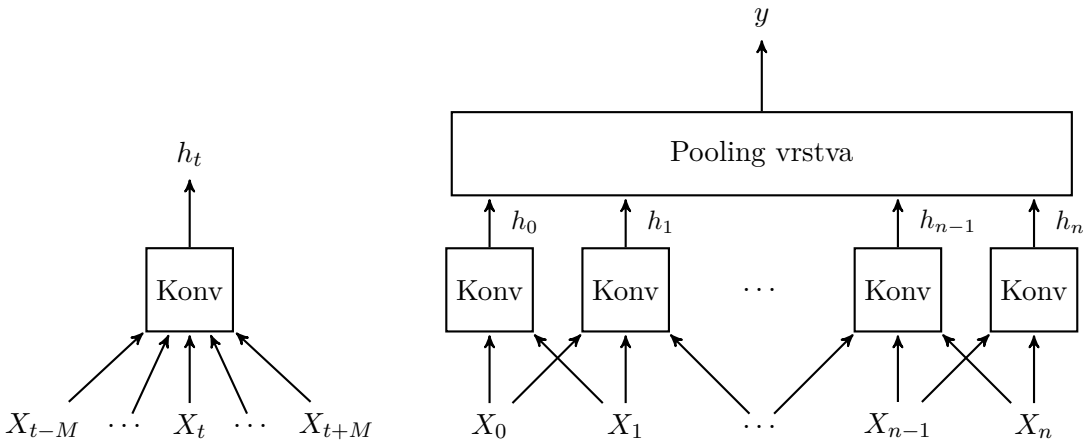
Problém *vzdálených závislostí* a „mizejícího“ gradientu lze vyřešit sítí typu *LSTM* (*Long short-term memory*), která byla navržena právě za tímto účelem [16]. Jedná se o speciální druh rekurentní neuronové sítě, která se liší rozdílnou strukturou buňky RNN z obrázku 2.1a. V buňce *LSTM* nalezneme obvykle 3 komponenty: vstupní, výstupní a paměťovou bránu. Jelikož existuje několik různých implementací, může některá komponenta přebývat nebo chybět. Vstupní brána určuje, jak se bude zacházet se vstupní hodnotou X_i, a analogicky výstupní brána určuje způsob výpočtu výstupní hodnoty h_i. Paměťová brána ovlivňuje to, jakým způsobem bude síť pracovat se svým vnitřním stavem. Buňka si do další iterace pamatuje kromě svého předchozího výstupu h_{t-1} také vnitřní stav c_{t-1}. Na vnitřní stav, oproti výstupu, jsou aplikované vždy pouze lineární operace jako sčítání a násobení. Díky této vlastnosti můžou sítě *LSTM* lépe zachytit *vzdálené* závislosti.

Výstupem sítě typu *LSTM* je levá kontextová reprezentace každého slova $\vec{h_t}$. V rámci zpracovávání přirozeného jazyka může obsahovat další užitečné informace i kontext následujících slov v sekvenci, tedy pravý kontext. Za tímto účelem byly navrženy sítě typu *Bi-LSTM*, které jsou složeny ze dvou *LSTM*. První počítá levou kontextovou reprezentaci slov $\vec{h_i}$ a druhá v opačném směru počítá pravou kontextovou reprezentaci $\overleftarrow{h_i}$. Výsledná kontextová reprezentace je zřetězení jednotlivých výstupů $h_i = \vec{h_i}\overleftarrow{h_i}$. Architektura sítě *Bi-LSTM* je znázorněna na obrázku 2.2.

Zástupcem sítí *Bi-LSTM* pro kontextovou reprezentaci slov je například model *ELMo*. Model *ELMo* využívá pro pravou a levou kontextovou reprezentaci několik vrstev sítí *LSTM* [33]. Výsledná reprezentace je zřetězení obou reprezentací, jak tomu je u *Bi-LSTM*. Počet vrstev je závislý na konkrétní úloze.

2.2.2 Konvoluční neuronové sítě

Konvoluční neuronové sítě jsou běžně využívané k vytváření *rysů* při zpracování obrazu. Lze je také využít k vytvoření kontextové reprezentace vět [18]. Rekurentní sítě popsané v předchozí kapitole dokáží jednoduše vytvářet kontextovou reprezentaci pro každé slovo vstupní věty. Reprezentaci kontextu celé věty potom tvoří vektor posledního slova, prvního slova, nebo jejich zřetězení. Při využití konvoluční sítě se obvykle vytváří pouze vektorová reprezentace celé věty.



(a) Konvoluční vrstva. (b) Paralelizace výpočtu kontextové reprezentace věty.

Obrázek 2.3: Architektura konvoluční neuronové sítě.

Konvoluční sítě se skládají z konvoluční a *pooling* vrstvy. Diskrétní 1D konvoluce mezi filtrem h a vstupním signálem x lze definovat jako

$$(x * h)[t] = \sum_{m=-M}^{M} x[t-m]h[m] \ . \tag{2.2}$$

Vstupní větu si můžeme představit jako zřetězení vektorů jednotlivých slov. Operace konvoluce lze potom chápat jako filtr, který pro každou pozici vektoru vypočítá novou hodnotu z jeho blízkého okolí. Následně můžeme vypočítat kontextovou reprezentaci každého slova posouváním více různých filtrů s různou maximální vzdáleností vždy o velikost vektorové reprezentace vstupních slov.

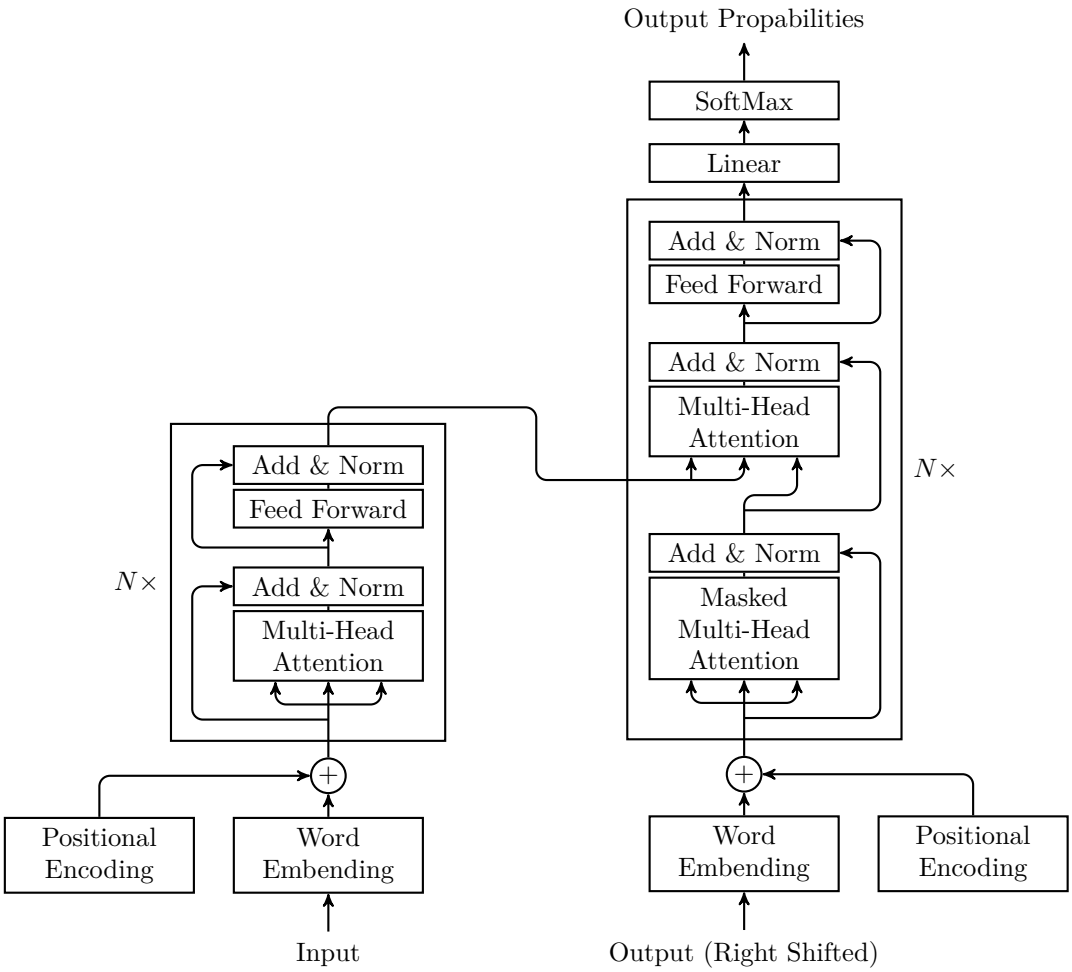
Konvoluční vrstva v neuronových sítích je běžná, plně propojená vrstva, která má na vstupu kromě vektorové reprezentace aktuálního slova také reprezentace slov v jeho blízkém okolí (vizte obrázek 2.3a). Stejně jako u rekurentních sítí tak můžeme posouváním sítě vytvořit pro každé slovo kontextovou reprezentaci. Zkoumané okolí se volí obvykle malé a pro zachycení dlouhých závislostí se přidává více konvolučních vrstev.

Jelikož věty mají proměnnou délku, využívá se *pooling* vrstva k vytvoření vektorové reprezentace věty s konstantní dimenzí. Jednou možnou implementací je použití funkce *max-pooling*. Pro každé slovo a dimenzi vektoru se vybere maximální hodnota, která se použije ve výsledné reprezentaci. Tímto se dosáhne konstantní velikosti vektoru, jehož dimenze je stejná jako počet použitých filtrů v konvoluci.

Konvoluční sítě lze využít také na úrovni slov místo celých vět [19]. Vstupem je tedy posloupnost znaků, která se převede na vektorovou reprezentaci podle vyhledávací tabulky. Tato tabulka je výrazně menší než pro slova. Aplikováním konvoluční sítě na každý znak slova se vytvoří jeho vektorová reprezentace.

2.2.3 Sítě typu transformers

V předchozích kapitolách byly představeny rekurentní a konvoluční neuronové sítě a jejich využití pro kontextovou reprezentaci slov. Použití rekurentních neuronových sítí se osvědčilo u mnoha problémů v oblasti zpracování přirozeného jazyka. Jejich hlavní nevýhodou je sekvenční vytváření reprezentace slovo po slově, které lze jen obtížně paralelizovat. Na

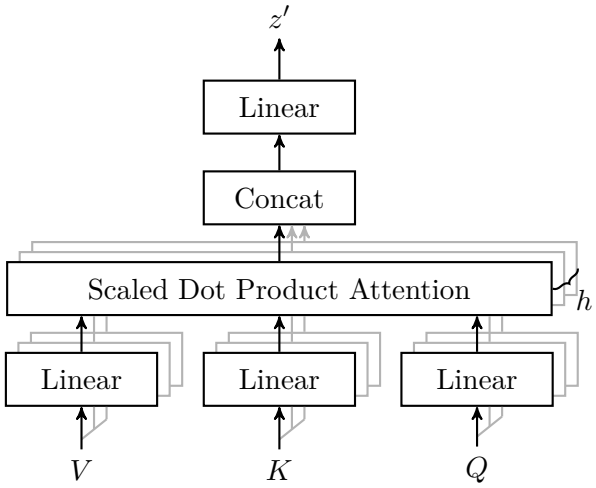


Obrázek 2.4: Architektura sítě typu *transformers*. Na obrázku vlevo se nachází modul kodéru, který vytváří interní reprezentaci vstupní věty. Vpravo se nachází komponenta dekodéru pro generování výstupní věty pomocí interní reprezentace.

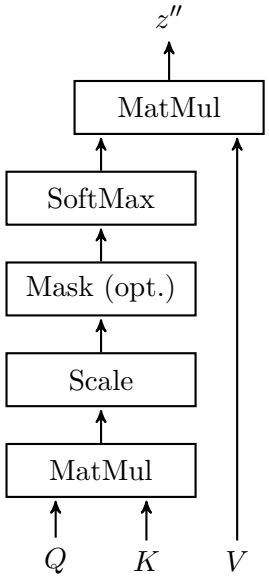
druhou stranu konvoluční neuronové sítě lze jednoduše paralelizovat a jsou vhodné pro zachycení lokálních závislostí. Pro zachycení *vzdálených* závislostí ale může být potřeba velkého množství vrstev (logaritmická složitost v závislosti na délce vstupu). V reakci na nedostatky těchto sítí vznikl návrh sítí typu *transformers*, které slibovaly jednoduchou paralelizaci a zachycení *vzdálených* závislostí s konstantní složitostí [41].

Sítě typu *transformers* vznikly pro řešení problémů jako strojový překlad, generování textu nebo predikci následujících slov a vět. Architektura modelu se skládá z kodéru a dekodéru (vizte obrázek 2.4). Kodér vytváří na základě vstupní věty, kterou chceme například přeložit, vnitřní reprezentaci slov. Tato reprezentace je následně využita při dekódování. Dekodér postupně slovo po slovu, počínaje počáteční značkou, generuje následující slovo, které se použije jako vstup pro další iteraci. Postup pokračuje, dokud nevygenerujeme koncovou značku.

Každé vstupní slovo je převedeno na vektorovou reprezentaci, například některým modelem z kapitoly 2.1. Ke každé vektorové reprezentaci je přidána informace o pozici slova ve větě. Oproti rekurentním a konvolučním sítím mají sítě typu *transformers* pevnou maximální délku vstupní sekvence.



(a) Vícehlavá funkce pozornosti *self-attention*.



(b) Detail bloku *Scaled Dot Product Attention*.

Obrázek 2.5: Blokové znázornění funkce *pozornosti*. Nalevo vidíme *vícehlavou funkci pozornosti* a napravo detail funkce *pozornosti* pro jednu z hlav.

V dnešní době vznikají různé předtrénované kodéry na velkých korpusech. Jako příklad lze uvést několik modelů[4]: GPT [35], GPT-2 [36], BERT [12], RoBERTa [27], ALBERT [23] nebo XLNet [45]. Pro kontextovou reprezentaci je důležitá komponenta kodéru, proto se zbytek kapitoly zaměří převážně na ni.

Kodér a dekodér

Kodér a dekodér je složen z konstantního počtu N vrstev. Každá vrstva je dále složena ze dvou podvrstev: vícehlavé funkce *pozornosti* a dopředné neuronové sítě. Kolem každé podvrstvy se provede propojení vstupu s výstupem a následná normalizace. To lze formálně vyjádřit jako

$$\text{layerNorm}(x + \text{subLayer}(x)) \ , \tag{2.3}$$

kde funkce *subLayer* reprezentuje konkrétní implementaci podvrstvy. Všechny výstupy vrstev a podvrstev modelu jsou stejné dimenze d_{model}. Dekodér je složením obdobný kodéru, kdy oběma podvrstvám ještě předchází podvrstva maskované vícehlavé funkce *pozornosti*, která má za úkol zaměřit pozornost dekodéru pouze na již vygenerovaná slova.

Funkce pozornosti

Funkce *pozornosti* (angl. *attention*) slouží k zaměření modelu na podstatné části věty. Funkce *pozornosti* využitá v sítích *transformers* je pojmenovaná *self-attention* a lze ji popsat jako překlad vstupního dotazu podle slovníku klíč-hodnota, kde dotaz, klíče, hodnoty i výstup jsou vektory reálných čísel (vizte obrázek 2.5b). Jednotlivé dotazy, klíče a hodnoty

[4]Seznam různých architektur: https://github.com/huggingface/transformers#model-architectures.

dostaneme vynásobením vstupní vektorové reprezentace slov třemi maticemi, přičemž určení jejich hodnot je předmětem učení. Každý dotaz se vynásobí se všemi klíči, výsledku se změní „měřítko“ a funkcí softmax se vypočítá váha dané kombinace dotaz-klíč. Hlavní myšlenkou poslední úpravy je přiřazení velmi nízké váhy ke klíčům, které nejsou v dané chvíli podstatné. Následně se provede vážený součet jednotlivých hodnot na základě dříve zmíněných vah. Maticově lze daný proces vyjádřit jako výraz (2.4) [41]

$$\text{attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \ , \tag{2.4}$$

kde Q je matice dotazů, K je matice klíčů, V je matice hodnot a d_k je dimenze vektorů dotazu a klíče.

Vícehlavá funkce pozornosti

Pro zvýšení výkonnosti jednoduché funkce *pozornosti* se využívá metoda *vícehlavé funkce pozornosti* (angl. *multi-head attention*). Jednoduchá funkce *pozornosti* se provádí vícekrát paralelně nad vstupem s různými lineárními projekcemi dotazu a klíče do prostoru o dimenzi d_k a projekcí hodnoty do prostoru o dimenzi d_v, vizte obrázek 2.5a. Výsledky jednotlivých h hlav se následně zřetězí a provede se projekce do prostoru o dimenzi modelu d_{model}. Toto umožňuje jednotlivým hlavám získat informace z různých reprezentací podprostoru na různých pozicích. Formálně můžeme výstup jednotlivých hlav vyjádřit vztahem (2.5) [41] a výslednou reprezentaci jako (2.6) [41], kde $W_i^Q, W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$ a $W^O \in \mathbb{R}^{h \cdot d_v \times d_{model}}$.

$$head_i = \text{attention}\left(QW_i^Q, KW_i^K, VW_i^V\right) \tag{2.5}$$

$$\text{multiHead}(Q, K, V) = \text{concat}\left(head_0, head_1, \ldots, head_h\right)W^O \tag{2.6}$$

Všechny parametry h, d_k, d_v a d_{model} jsou závislé na konkrétní implementaci, přičemž když se rozumně sníží dimenze vektorů, se kterou počítá každá hlava, tak se dosáhne podobné výpočetní náročnosti jako při použití jedné hlavy s plnou dimenzí vektorů d_{model}.

Transformers-XL

Nevýhodou sítí typu *transformers*, oproti rekurentním a konvolučním sítím, je práce se vstupem o pevné délce. Kratší věty, než je požadovaný počet slov, lze jednoduše doplnit vyplňujícími tokeny a zavést masku, která zaměří výpočet sítě pouze na tokeny vstupní věty. U vytváření reprezentace pro delší věty dochází k problému, jak větu rozdělit do segmentů o konstantní délce, a k problému chybějícího kontextu z předchozího segmentu. Tento problém řeší rekurentní mechanismus *Transformers-XL* [11], kde *XL* odkazuje právě na velmi dlouhé věty (angl. *eXtra Long*).

Samotná architektura se od dříve popsané téměř neliší, pouze dochází k záměně absolutního pozičního kódování za relativní a přidání paměti. Vstupní věta je nejdříve rozdělena do segmentů o pevné délce, kde je poslední segment vyplněn zprava (resp. první segment zleva, podle implementace) speciálními tokeny. Výpočet reprezentací pro jednotlivé segmenty se provádí rekurentně, kdy se vždy do paměti uloží výstupní stav každé vrstvy. V dalším segmentu se využije paměť ve funkci pozornosti *self-attention*. Vstupní dotazy Q zůstávají stejné, jak byly popsány v předchozí kapitole, ale ke vstupním klíčům K a hodnotám V se zřetězí odpovídající vektorové reprezentace dané vrstvy z předchozího segmentu, které jsou

uloženy v paměti. Tyto hodnoty jsou následně zobrazeny do požadované dimenze. Změnou pozičního kódování a přidáním paměti je elegantně vyřešen problém omezené délky vstupu sítí typu *transformers*.

2.3 Shrnutí

V této kapitole byly představeny základní modely využívané při analýze přirozeného jazyka. První část kapitoly se věnuje vektorové reprezentaci slov, která by byla schopna zachytit syntaktické podobnosti slov. Představili jsme si model *Word2Vec* založený na neuronových sítích a model *GloVe* založený na statistické metodě matic společných výskytů.

Druhá část kapitoly pojednává o využití neuronových sítí k reprezentaci kontextu vět. Vstupem neuronových sítí jsou předtrénované vektorové reprezentace slov popsané v první části kapitoly. Pro zachycení informací o kontextu ve větě se osvědčily rekurentní a konvoluční neuronové sítě. Nevýhodou rekurentních sítí je složitá paralelizace. Naopak konvoluční neuronové sítě lze jednoduše paralelizovat, ale pro zachycení *vzdálených* závislostí je potřeba logaritmický počet konvolučních vrstev. Jako řešení těchto problémů vznikly sítě typu *transformers* založené na funkci *pozornosti* zvané *self-attention*.

Kapitola 3

Extrakce vztahů mezi pojmenovanými entitami

S rozmachem internetu se zpřístupnily nové rozsáhlé zdroje stále přibývajících informací v podobě novinových nebo výzkumných článků, elektronických knih, webových stránek, fór nebo e-mailů. Všechny tyto zdroje jsou v podobě nestrukturovaného textu, který je přirozený pro komunikaci mezi lidmi, ale pro strojové zpracování není příliš vhodný. Proto se vyvíjí úsilí vytvořit automatizované vyhledávání nových informací v textu a jejich reprezentování ve strukturované podobě.

Nově získané informace můžou být potencionálně využity například při zdokonalení strojového překladu, vyhledávání na internetu, vytváření sémantických grafů nebo při komunikaci s umělou inteligencí v přirozeném jazyce. Z mnoha různých úloh v této oblasti se budou následující kapitoly zabývat rozpoznáváním pojmenovaných entit v textu a klasifikací vztahu mezi jednotlivými entitami.

3.1 Rozpoznávání pojmenovaných entit

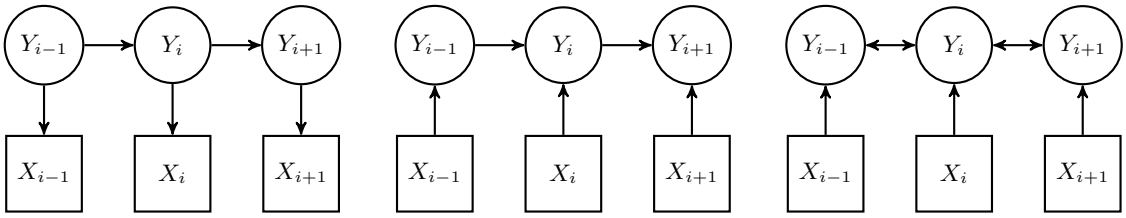
Problém extrakce pojmenovaných entit spočívá v identifikaci frází ve větě, které např. odpovídají jménům osob, organizací nebo míst. Tyto kategorie podporují snad všechny dnešní systémy. Dalšími kategoriemi můžou být produkty, věci nebo specializované kategorie jako umělci a politici. Zvláštní skupinu pojmenovaných entit tvoří data a čísla. Jako příklad uvažujme následující větu:

„Karel Čapek (Osoba) získal svůj doktorát na fakultě Univerzity Karlovy (Organizace) v Praze (Lokace).“

V této větě se nacházejí tři pojmenované entity. První je osoba „Karel Čapek“, další je název organizace „Univerzita Karlova“ a poslední rozpoznanou entitou je lokace „Praha“.

Cílem této úlohy je pouze identifikace pojmenovaných entit ve větě, případně jejich zařazení do některé z kategorií, ale bez určení sémantiky. Takže jsme z textu rozpoznali, že Karel Čapek je nějaká osoba, ale nevíme, že se jedná o spisovatele Karla Čapka. Tímto problémem se dále zabývá úloha *entity linking*, která každé pojmenované entitě přiřazuje položku ze znalostní báze[1].

[1] Znalostní báze je databáze obsahující např. informace o objektech reálného světa a vztazích mezi nimi.



(a) Skrytý Markovův model. (b) Markovův model s maximální entropií. (c) Podmíněné náhodné pole.

Obrázek 3.1: Grafická struktura modelů pro problém sekvenčního značení.

Extrakce pojmenovaných entit je důležitá fáze při získávání informací, například při extrakci vztahů z vět, zefektivnění vyhledávacích algoritmů nebo automatické kategorizaci článků. Společně s úkolem POS (angl. *part-of-spech*) patří extrakce pojmenovaných entit do skupiny problému sekvenčního značení, kdy se jednotlivým slovům ve větě přiřazují lingvistické značky.

Pro problém extrakce pojmenovaných entit se značka skládá z pozice slova v rámci pojmenované entity, typu pojmenované entity, případně další volitelné vlastnosti. Pro označení pozice slov v rámci entity se používají obvykle značky:

- B – první slovo entity,
- I – slovo entity, které není reprezentované již jinou značkou,
- E (L) – poslední slovo entity,
- O – slovo stojící mimo jakoukoliv entitu ve větě,
- U – jednoslovná entita.

Podle použité množiny značek existuje několik modelů [20]: *IO*, *BIO*, *IEO*, *BIEO* a *BILOU*. Jednotlivé modely mají stejnou vyjadřovací sílu, ale díky explicitnímu značení lze model *BILOU* „snáze“ naučit [37].

V dalších kapitolách bude nejdříve popsána metoda *podmíněných náhodných polí*, která v kombinaci s neuronovými sítěmi využívá mnoho současných řešení úlohy, a následně budou popsány aktuální řešení v oblasti rozpoznávání pojmenovaných entit.

3.1.1 Podmíněné náhodné pole

Metoda *podmíněných náhodných polí* (angl. *conditional random field*) [21], dále *CRF*, se postupem času stala velmi populární při řešení problému sekvenčního značení, a to hlavně kvůli své jednoduchosti a účinnosti. Mnoho ze současných řešení v oblasti rozpoznávání pojmenovaných entit využívá *CRF* v kombinaci s různými neuronovými sítěmi [17, 33, 40].

Tato metoda vznikla jako řešení nedostatků *Markovova skrytého modelu* a *Markovova modelu s maximální entropií* [28] při zachování jeho výhod. Nevýhodou *Markovova skrytého modelu* je nutnost k určení pravděpodobnosti daného značení generovat všechny možné kombinace značení. Naopak *Markovův model s maximální entropií* trpí takzvaným problémem dvojího značení (*bias label problem*).

Dále v kapitole budeme náhodnou veličinu sekvence, která má být označena, značit jako X a náhodnou veličinu již označené sekvence jako Y. Pro náš konkrétní případ chápeme

X jako větu složenou ze sekvence slov nebo jejich rysů a Y jako konkrétní označení pojmenovaných entit. V rámci této metody se tvoří podmíněný model $p(X|Y)$ z dvojice věty a jejího značení, aniž by se výslovně modelovala pravděpodobnost $p(X)$. *CRF* lze formálně definovat následovně [21]:

Definice 3.1.1. Mějme neorientovaný graf $G = (V, E)$ takový, že vektor $Y = (Y_v)_{v \in V}$ je indexován vrcholy grafu G. Potom dvojice (X, Y) je *podmíněné náhodné pole*, když je podmíněna náhodnou veličinou X a náhodná veličina Y se řídí Markovskými vlastnostmi s ohledem na graf G: $P(Y_v|X, Y_w, w \neq v) = P(Y_v|X, Y_w, w \sim v)$, kde $w \sim v$ znamená $\{w, v\} \in E$.

Obecně můžeme mít libovolný graf G, ale pro potřeby extrakce pojmenovaných entit si vystačíme s jednoduchým řetězcem, neboli $G = (\{1, \ldots, m\}, \{\{i, i+1\} \mid 1 \leq i \leq m-1\})$. To znamená, že každé značení konkrétního slova je ovlivněno značením předchozího a následujícího slova. Tyto hrany grafu nám pomáhají dodržet některé pevně dané omezení, například po značce *B-ORG* nemůže následovat značka *E-PER*.

Pravděpodobnost, že značení vstupní věty x odpovídá značení y, můžeme vypočítat pomocí vztahu (3.2) [21], kde $\theta = \{\lambda_1, \lambda_2, \ldots, \lambda_n; \mu_1, \mu_2, \ldots, \mu_m\}$ jsou parametry, které se snažíme optimalizovat, značením $y_{|S}$ je myšlena podsekvence y odpovídající podgrafu S, hodnotící funkce f_k pro hrany a hodnotící funkce g_k pro vrcholy grafu.

$$s_\theta(x, y) = \sum_{e \in E} \sum_k \lambda_k f_k(e, y_{|e}, x) + \sum_{v \in V} \sum_k \mu_k g_k(v, y_{|v}, x) \tag{3.1}$$

$$p_\theta(y|x) = e^{s_\theta(x,y)} \tag{3.2}$$

Funkce f_k a g_k jsou pevně dané a funkce f_k nám obvykle hodnotí dvě po sobě jdoucí značky a funkce g_k vhodnost přidělené značky y_i k *rysům* slova x_i.

Při učení modelu nad datovou sadou $D = \{(x, y)|x \in X, y \in Y\}$ se snažíme úpravou parametrů θ maximalizovat pravděpodobnost $p_\theta(y|x)$ každé položky v datové sadě.

3.1.2 Související práce

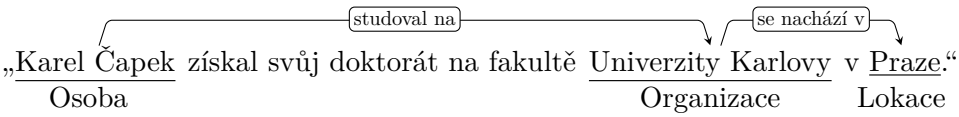
Nejrozšířenější přístup v současných řešeních je využití kombinací sítě *Bi-LSTM* a modelu *CRF* [22]. Rekurentní síť vytvoří kontextovou reprezentaci každého slova a model *CRF* predikuje na základě reprezentace konkrétního značení. Vylepšením tohoto přístupu je model *FLAIR* složený ze dvou rekurentních vrstev [1]. První vrstva má na vstupu místo celých slov jednotlivé znaky. Vnitřní reprezentace každého slova je vytvořena spojením vektorové reprezentace prvního a posledního písmene slova. Druhá vrstva je již zmíněná kombinace *Bi-LSTM* a *CRF*.

Předtrénovaný model *BERT* dosahuje dobrých výsledků i bez použití *CRF* jako výstupní vrstvy [12]. Ukázalo se jako přínosné kombinovat sítě *Bi-LSTM* a *FLAIR* s dalšími sítěmi jako *BERT* nebo *ELMo* a modelem *CRF* [17, 33].

Zajímavým přístupem je vytvoření nového modelu sítě typu transformer [4]. Nový kodér se liší od původního, popsaného v kapitole 2.2.3, jen v několika drobnostech. Vstupní reprezentace slov je založena na předtrénované konvoluční síti. Funkce *pozornosti* se použije pouze na předcházející nebo následující slova ve větě, a tím se podobá sítím typu *Bi-LSTM*. Dalším rozdílem je použití normalizace na vstupu každé podvrstvy namísto jejího výstupu. Kontextovou reprezentaci slov vytvoříme spojením výsledků kodérů pro pravý a levý kontext.

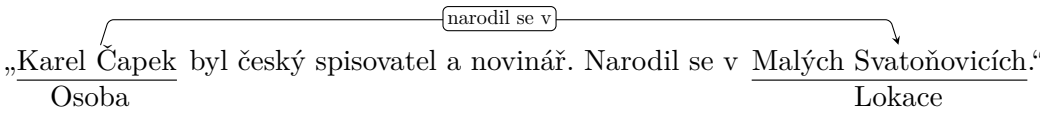
3.2 Extrakce sémantických vztahů z textu

Cíl extrakce vztahů spočívá v určení sémantických vazeb mezi pojmenovanými entitami ve větě. Jedná se o zásadní problém v oblasti získávání informací z textu a zpracovávání přirozeného jazyka. Opět pro demonstraci lze uvést větu z předchozí kapitoly



Z této věty můžeme identifikovat dva vztahy mezi pojmenovanými entitami: Karel Čapek studoval na Univerzitě Karlově a Univerzita Karlova se nachází v Praze. Na základě těchto vztahů lze odpovídat na další otázky, jako například „Ve kterém městě studoval Karel Čapek?“ Konkrétní aplikací extrakce vztahů může být hledání zmínek o fúzi firem, nově objevených interakcí proteinů publikovaných ve vědeckých článcích nebo automatizované vytváření znalostních bází.

Obecně se nemusí pojmenované entity nacházet pouze v té samé větě, ale i v různých větách dokumentu, například:



Zachycení vztahů napříč celým dokumentem je výrazně obtížnější, proto se většina dnešních řešení omezuje pouze na extrakci vztahu v rámci jedné věty.

Extrakce vztahů lze dále rozdělit podle dostupných dat k učení modelu. Když jsou k dispozici úplná anotovaná data, tak se jedná o tradiční extrakci vztahů. V tomto případě je hlavním cílem klasifikovat vztah do jedné z tříd, včetně speciální třídy pro neznámý, nebo žádný vztah.

V jiném případě sice existují anotovaná data, ale zajímají nás vztahy, které v trénovací datové sadě chybí [25], nebo mají velmi malé množství vzorků [14]. V takovém případě se model natrénuje na dostupné trénovací sadě, která neobsahuje množinu požadovaných vztahů. Při klasifikaci má model k dispozici kromě věty obsahující pojmenované entity také reprezentativní vzorky nových, předtím nikdy neviděných tříd vztahů. Reprezentativním vzorkem může být např. pár vět obsahujících daný vztah mezi pojmenovanými entitami, pokud je máme k dispozici. Model musí být schopen na základě reprezentativního vzorku určit, zda mezi pojmenovanými entitami existuje daný vztah.

Pokud je k dispozici pouze neanotovaná datová sada nebo chybí úplně, tak se extrakce vztahů nejčastěji omezí na nalezení vztahu bez jeho klasifikace. Vztah se následně vyjádří pomocí fráze z věty. Když budeme opět uvažovat první příklad, tak bychom vztah mezi entitou „Karel Čapek“ a „Univerzita Karlova“ vyjádřili pomocí fráze „získal doktorát na“, aniž bychom uvažovali sémantiku daného vztahu. Speciálním případem, který lze do této kategorie zařadit, je úloha *Open Information Extraction* [6]. Při řešení této úlohy se v textu nehledají ani pojmenované entity, pouze se identifikuje přísudek a podmět, případně předmět a příslovečné určení. Podmět s předmětem suplují pojmenované entity a přísudek vztah mezi nimi. Samotná extrakce je při absenci anotovaných dat založena na ručně napsaných pravidlech, závislých na syntaxi konkrétního přirozeného jazyka, nebo rozšíření počáteční sady pravidel pomocí trénovacích dat s využitím např. metody *bootstrapping*.

Obvykle se získávají binární vztahy, tzn. vztahy právě mezi dvěma entitami. Obecně lze ale v textu hledat libovolné n-ární vztahy, s čímž se nejčastěji setkáme u úlohy *Open Information Extraction*.

3.2.1 Klasifikace vztahů mezi pojmenovanými entitami

Tato práce se zabývá tradiční klasifikací binárních vztahů mezi pojmenovanými entitami v rámci jedné věty. Formálně můžeme problém definovat následovně.

Definice 3.2.1. Mějme množinu známých vztahů $R = \{r_1, r_2, \ldots, r_n\}$ obsahující speciální neznámý vztah $\emptyset \in R$ a větu $S = (w_1, \ldots, w_i, \ldots, w_j, \ldots, w_k, \ldots, w_l, \ldots, w_m)$, kde w_i jsou slova, $e_1 = (w_i, \ldots, w_j)$ a $e_2 = (w_k, \ldots, w_l)$ jsou pojmenované entity. Pro každý vztah $r \in R$ chceme určit pravděpodobnost $P(r|S, \langle e_1, e_2 \rangle)$.

Pokud máme k dispozici ohodnocená data, tak se nabízí použít různé metody strojového učení. Tradiční přístupy pro identifikaci vztahů jsou založeny na *rysech* a *jádrech* [3]. U prvního přístupu jsou jednotlivá slova obohacena například o POS značky, vzdálenosti od entit, druh pojmenovaných entit nebo cestu mezi entitami v syntaktickém stromu. Tyto informace jsou reprezentované v podobě vektoru, podle kterého se provádí klasifikace.

Metody založené na *jádrech* určují podobnost mezi dvěma objekty, které můžou být slovy, větami nebo syntaktickými stromy. Podobnost se vypočítá na základě počtu stejných podsekvencí (podgrafů) v obou objektech. Tyto přístupy jsou závislé na externích nástrojích, které vytvářejí dodatečné informace z textu nebo vytváří syntaktické stromy. Externí nástroje mají svou míru chybovosti, a tím ovlivňují účinnost metod.

Neuronové sítě dokážou vytvořit interní reprezentaci věty, zachycující sémantické a syntaktické závislosti, bez nutnosti použití externích nástrojů. Některé modely využívají navíc externí nástroje pro zvýšení výkonnosti. V minulosti se pro tuto úlohu zvláště osvědčily rekurentní a konvoluční neuronové sítě nebo kodéry ze sítí typu transformer.

3.2.2 Související práce

V následující kapitole budou rozebrány současné přístupy, které pro svoji funkcionalitu nepotřebují dodatečné znalosti nebo externí nástroje. Některé modely můžou využívat externí nástroje pro zvýšení výkonnosti, ale nejsou na nich závislé.

Rekurentní neuronové sítě

Řešení s rekurentními sítěmi využívají druh sítě *Bi-LSTM* k vytvoření interní reprezentace jednotlivých slov ve větě. Tato interní reprezentace může být pro dosažení lepších výsledků obohacena o dodatečné informace zmíněné již dříve [48]. Z interní reprezentace se vytvoří pomocí různých implementací funkcí *pozorností* reprezentace věty, která slouží ke klasifikaci [50]. Obvykle je funkce *pozornosti* založená na váženém průměru, kdy jsou váhy předmětem učení. Zajímavé je využití funkce *pozornosti* z kodéru sítí transformer (vizte kapitolu 2.2.3) pro zvýšení výkonnosti tohoto modelu. Funkce je aplikována jako první vrstva na vektorové reprezentaci vstupní věty ještě před sítí *Bi-LSTM* [24].

Konvoluční neuronové sítě

Modely využívající konvoluční sítě byly inspirovány tradičními metodami založenými na *rysech*. Ke každé vektorové reprezentaci je přidána informace o relativní pozici od každé

pojmenované entity, případně další informace jako POS značky. Následně je za pomoci konvoluční neuronové sítě vytvořena reprezentace celé věty. K reprezentaci věty může být přidána ještě lexikální reprezentace, která obsahuje vektorové reprezentace obou entit a jejich okolí, nebo jiné informace z externích nástrojů [46]. Jiný přístup ukázal užitečnost vytvoření reprezentace pro jednotlivé entity za využití funkce *pozornosti* a její připojení ke kontextové reprezentaci věty [38]. Funkce pracuje se vstupní vektorovou reprezentací věty a pomáhá zachytit, které části věty mají vliv na určení vztahu v závislosti na konkrétní entitě. Další zlepšení může být dosaženo použitím různých funkcí *pozornosti* na více úrovních jako na vstupní vektorové reprezentaci a vstupu *pooling* vrstvy [43].

Sítě typu transformers

S příchodem sítí typu transformer bylo jen otázkou času, kdy se ukáže jejich potenciál i v úloze rozpoznávání vztahů mezi pojmenovanými entitami. Dosavadní řešení využívají metody *dotrénování* (angl. *fine-tuning*), kdy se využije již na jiné úloze předtrénovaný kodér, konkrétně systém *BERT*, a nad ním se vybuduje jednoduchá neuronová síť. Při dotrénování postačí pouze pár epoch na menší datové sadě, kdy se učí hlavně přidané poslední vrstvy. Pro úlohu extrakce vztahů se nad kodérem navrhne jednoduchá plně propojená vrstva pro klasifikaci, využívající vektorovou reprezentaci celé věty a jednotlivých slov entit. Nezávisle na sobě se ukázala výhodnost přidání speciálních značek do vstupní věty pro ohraničení pojmenovaných entit [5, 44].

3.3 Shrnutí

Kapitola pojednává o úloze extrakce vztahů z textu s využitím strojového učení. Úlohu lze rozdělit do dvou částí: rozpoznání pojmenovaných entit a klasifikace vztahu mezi nimi. Obě podúlohy byly definovány a byla také popsána některá aktuální řešení. Při rozpoznávání pojmenovaných entit se osvědčilo využití rekurentních sítí *Bi-LSTM*, kodérů sítí typu *transformers* a metody *podmíněných náhodných polí*. V rámci klasifikace vztahů mezi pojmenovanými entitami bylo rozebráno využití rekurentních, konvolučních a sítí typu *transformers*.

Kapitola 4

Návrh a implementace

V následující kapitole jsou představeny požadavky a návrh systému. Cílem práce je vytvořit systém pro extrakci vztahů mezi pojmenovanými entitami, který zpracovává nově získaná data z internetu. Samotná klasifikace vztahů je založena na výsledcích aktuálního výzkumu v oblasti zpracování přirozeného jazyka. Hlavními požadavky na systém jsou snadná přenositelnost, modularita a snadné rozšíření budoucí funkcionality.

Pro dosažení požadavků na systém byl navržen jednoduchý *framework*, kdy si uživatel pomocí konfiguračního souboru zvolí množinu požadovaných formátů vstupních souborů, sekvenci jednotlivých modulů pro zpracování, jejich nastavení a formát uložení výsledných dat. Pro účely této práce byly navrženy a implementovány dva moduly pro zpracování, a to rozpoznávání pojmenovaných entit (kapitola 4.3) a klasifikace vztahů mezi nimi (kapitola 4.4). Blokové schéma návrhu můžete vidět na obrázku 4.1.

Systém je implementován v jazyce *Python 3*, kdy lze systém rozšířit o další moduly implementováním potřebných rozhraní. Jako základní podporované formáty byly zvoleny prostý text, HTML, WARC a MG4J. Pro rozpoznání pojmenovaných entit se využije volně dostupný systém a pro klasifikaci vztahů je navržen vlastní model. Výsledky klasifikace jsou uloženy ve formátu N-Triples.

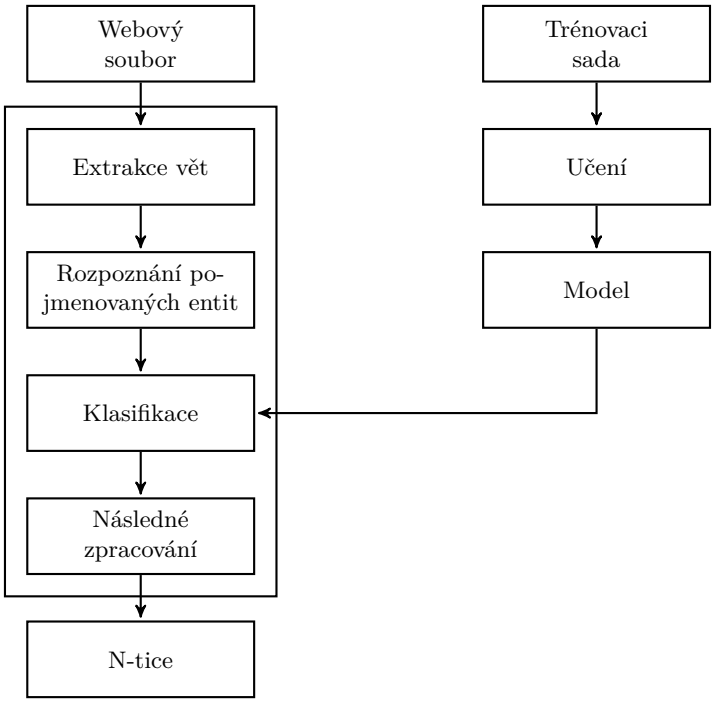
4.1 Konfigurační soubor

Jak bylo v úvodu kapitoly zmíněno, systém byl navržen, aby byl co nejvíce modulární a šel v budoucnu rozšířit o další funkcionalitu. Konfigurační soubor slouží k vytvoření a nastavení celého procesu, od načtení vstupních souborů až po jejich uložení.

Pro strukturu konfiguračního modelu lze vytvořit svůj vlastní formát, nebo využít některý z již existujících formátů, např. JSON, YAML, nebo XML. Pro tuto práci byl zvolen formát JSON. Struktura konfiguračního souboru je definována pomocí tzv. JSON schématu (resp. jeho alternativou vhodnou pro použití v jazyce *Python*). Díky použití jazyka pro definování struktury lze následně k její validaci a zpracování souboru využít existujících knihoven.

V konfiguračním souboru lze vybrat nastavení a konkrétní implementace modulů, které budou popsány v následujících kapitolách. Tato kapitola se hlavně zaměří na zpracování konfiguračního souboru a načítání modulů. Návrh konfiguračního souboru byl inspirován systémem BUbiNG[1] [8]. Systém BUbiNG dovoluje parametrizovat některé své části, např.

[1] Webový prohledávací modul (ang. *web crawler*) pro distribuované stahování webu, který je využíván výzkumnou skupinou Znalostních technologií FIT VUT v Brně.



Obrázek 4.1: Návrh systému pro extrakci vztahů mezi pojmenovanými entitami z webových stránek.

zpracování stránek nebo filtraci, pomocí zadání konstruktoru třídy s parametry do konfiguračního souboru. Následně je dynamicky načtena požadovaná implementace za běhu programu. Tento přístup byl převzat i do systému pro extrakci vztahů.

Každý modul je v konfiguračním souboru reprezentován strukturou s povinnou položkou „class“ a volitelnou položkou „kwargs“. Intuitivně první položka udává název konkrétní implementace modulu a volitelná položka je slovník argumentů. Pokud není pole pro argumenty uvedeno, načte se třída bez argumentů.

Samotné zpracování konfiguračního souboru je rozděleno do dvou fází: *startup* a *runtime*. První fáze provede načtení konfiguračního souboru, jeho validaci a doplnění nepovinných položek na výchozí hodnoty. V druhé fázi dojde k načtení jednotlivých modulů. Jazyk Python patří mezi dynamické jazyky, proto dovoluje jednoduše načítání modulů za běhu programu. Moduly jsou rozděleny na tři druhy: načtení vstupních souborů, analýza textu a uložení výsledků. Každý modul musí implementovat požadované rozhraní podle své funkcionality a musí být uložen v místě, kde ho následně bude program načítat. Každý druh modulu má svůj vlastní podadresář, kde se musí nacházet všechny jeho implementace. Tím je zamezeno, aby mohl uživatel načítat třídy z libovolných knihoven.

Systém může být v budoucnu jednoduše rozšiřován jak o nové podporované vstupní, tak výstupní formáty. Také lze využít nové modely pro rozpoznávání pojmenovaných entit, nebo klasifikaci vztahu, případně přidávat další moduly pro zpracování přirozeného jazyka.

4.2 Extrakce vět

Úkolem modulů pro zpracování vstupních souborů je extrakce textu a jeho rozdělení do vět, přičemž si může uživatel pomocí konfiguračního souboru zvolit sadu modulů pro různé

formáty vstupních souborů. Systém lze v budoucnosti jednoduše rozšířit o další podporované formáty. Postačí implementovat jednoduché rozhraní se dvěma metodami sloužícími jako predikáty. První predikát rozhodne, zda dokáže modul zpracovat soubor s daným jménem. Druhý predikát obdobně určí schopnost zpracovat soubor na základě typu internetového média (tzv. MIME typu). První predikát pomůže vybrat správný modul pro zpracování vstupních souborů, zatímco druhý se využije při zpracování záznamu např. v souborech WARC.

Při zpracování vstupního souboru každý z modulů postupně aplikuje predikát na jméno souboru, zda je schopný soubor s daným jménem zpracovat. V případě možného konfliktu, kdy by existovalo více modulů schopných soubor zpracovat, se zvolí poslední uvedený modul. Uživatel si tedy může pořadím modulů určit jejich prioritu. Například první modul bude obecně zpracovávat všechny soubory s příponou „.warc“. Druhý modul v pořadí může zpracovávat všechny soubory s předponou „bubing“ a příponou „.warc“, jelikož systém BUbiNG může do hlaviček přidávat své vlastní informace, například o duplicitě stránek. Tím máme zaručeno, že všechny soubory WARC začínající prefixem „bubing“ zpracuje druhý modul a ostatní soubory zase první modul.

4.2.1 Podporované formáty

V rámci této práce byly implementovány moduly pro zpracování souborů ve formátu prostého textu, HTML, WARC a MG4J. Pro zpracování prostého textu se využije knihovna *nltk*, která rozdělí text na samostatné věty. U souborů ve formátu HTML se nejdříve provede extrakce textu mezi jednotlivými značkami, který je následně zpracován stejně jako prostý text. Dále se kapitola podrobněji zaměří na zpracování souborů WARC a MG4J.

Soubory ve formátu WARC jsou kolekcí webových stránek, obvykle stažených webovým prohledávacím modulem (angl. *web crawler*). Běžný záznam se skládá ze tří částí: hlavička WARC, hlavička HTML a stažený obsah stránky. Jednotlivé záznamy jsou od sebe odděleny dvojicí prázdných řádků (ASCII znaky CR+LF). WARC hlavička obsahuje různé informace o staženém souboru jako odkaz URL, čas zahájení stahování, haš stránky, nebo typ souboru. Při zpracování nás z hlavičky WARC nejvíce zajímá položka *Content-Type*, obsahující typ internetového média (tzv. MIME typ). Na jeho základě se vybere jeden z předchozích zmíněných modulů a z stažené stránky se extrahují jednotlivé věty. Tedy pro zpracování záznamů ve WARC souboru a samostatných souborů se používají stejné moduly.

Posledním podporovaným formátem je MG4J. Tento formát je vstupem pro indexaci textu stejnojmenným nástrojem a také se jedná o výstup nástrojů pro zpracovávání přirozeného jazyka výzkumné skupiny Znalostních technologií. O této sadě nástrojů bude blíže pojednávat kapitola 4.3.4. Soubor je podobně jako relační databáze rozdělen do sloupců, které jsou odděleny znakem tabulátoru. Každý ze sloupců potom reprezentuje některý z výstupů syntaktické a sémantické analýzy. Tuto strukturu porušují pouze speciální řídící informace, které oddělují jednotlivé záznamy, odstavce a věty, nebo poskytují dodatečné informace o titulku a odkazu webové stránky. Navíc oproti předchozím formátům, ze kterých se pouze extrahovaly jednotlivé věty, se předají dalším modulům i všechny ostatní získané informace, mezi kterými jsou i rozpoznané pojmenované entity z textu.

Systém lze implementací zmíněného rozhraní jednoduše rozšířit o další formáty, jako např. formát PDF. Pro jeho použití následně stačí uvést název třídy, případně další parametry, do konfiguračního souboru mezi ostatní moduly pro zpracování vstupních souborů.

4.3 Rozpoznání pojmenovaných entit

Pro zpracování vstupního textu a rozpoznání pojmenovaných entit bude využit již některý z veřejně dostupných nástrojů. Dále v kapitole budou uvedeny čtyři nástroje se stručným popisem. Pro potřeby práce se počítá s využitím nástrojů *Corpproc*, nebo *SpaCy*. Pokud je vstupní soubor ve formátu mg4j (výstup nástroje *Corpproc*), jsou pojmenované entity načteny rovnou ze souboru pomocí modulu pro extrakci vět (vizte kapitolu 4.2). Dále bude popsáno několik nástrojů pro syntaktické zpracování textu, zvláště se zaměřením na rozpoznávání pojmenovaných entit.

4.3.1 Stanford Named Entity Recognizer

Stanford Named Entity Recognizer[2] je součástí projektu *CoreNLP* implementovaného v jazyce Java. Model podporuje rozpoznání 3 entit (*person*, *organization*, *location*) a je trénován na anglické datové sadě *CoNLL 2003*. Architektura modelu je založena na *podmíněných náhodných polích* [13]. I když je model implementován v jazyce Java, tak existuje mnoho rozhraní pro jiné jazyky, jako například JavaScript, PHP, Ruby nebo Python.

4.3.2 NLTK

Platforma *NLTK*[3] je určena pro vytváření programů v jazyce *Python* zpracovávajících přirozený jazyk. Jedná se o volně dostupný *open-source* projekt. Nástroj nabízí velkou škálu modulů pro klasifikaci, tokenizaci, syntaktické a sémantické zpracování, rozhraní pro jiné knihovny z oblasti zpracovávání přirozeného jazyka a mnohé další.

Součástí nástroje je také rozpoznávání 9 běžných pojmenovaných entit s použitím modelu značení *BIO*. *NLTK* také obsahuje rozhraní pro použití balíčků *Stanford Named Entity Recognizer* napsaných v jazyce Java. Dalším modulem je také extrakce vztahů z textu, kdy je vztahem chápána sekvence slov mezi entitami ve větě. Tento přístup lze spíše zařadit do kategorie *open infromation extraction*.

4.3.3 SpaCy

SpaCy[4] je *open-source* knihovna pro zpracovávání přirozeného jazyka v jazyce Python. Cílem vývojářů bylo vytvořit optimalizovaný nástroj pro jednoduchý vývoj aplikací, který v porovnání s *CoreNLP* a jinými nástroji dosahuje nejrychlejšího zpracování [10].

Nástroj podporuje tokenizaci textu, vektorovou reprezentaci slov, POS značení, syntaktické stromy, rozpoznávání pojmenovaných entit a různé další moduly. Pro rozpoznávání pojmenovaných entit je dostupných několik modelů v různých jazycích, ale také je možné natrénovat si svůj vlastní. Model pro angličtinu je trénován na datové sadě *OntoNotes 5*[5] a obsahuje 24 různých druhů entit.

4.3.4 Corpproc

Corpproc[6] je sada nástrojů pro stažení, zpracování a indexaci rozsáhlých textových korpusů využívaných výzkumnou skupinou Znalostních technologií na Fakultě informačních techno-

[2] https://nlp.stanford.edu/software/CRF-NER.html
[3] https://www.nltk.org/
[4] https://spacy.io/
[5] https://catalog.ldc.upenn.edu/LDC2013T19
[6] http://knot.fit.vutbr.cz/corpproc/

logií Vysokého učení technického v Brně. Nástroje jsou uzpůsobeny pro distribuované zpracování více počítači. Celé zpracování se skládá z několika kroků: stažení dat, vertikalizace souborů, deduplikace, syntaktické zpracování, sémantické obohacení textu a indexace.

Nástroje zpracovávají stažená data ve formátu WARC, HTML nebo Universal Verticalization Format[7]. Součástí jsou nástroje pro automatizované stažení dokumentů z různých zdrojů jako Wikipedie, CommonCrawl nebo RSS.

Stažené dokumenty se musí převést do vertikálního formátu. Vertikální soubor obsahuje na každém řádku jedno slovo, číslo, oddělovač, nebo řídící XML značku. Značky nám poskytují dodatečné informace, jako například začátek a konec vět, odstavců, nebo dokumentu. Ve fázi deduplikace jsou z vertikálních souborů odstraněny duplicitní odstavce. Deduplikace se opět může provádět distribuovaně napříč několika počítači.

Převést vstupní dokumenty do vertikálního formátu je důležité pro další zpracování, kdy se ke každému tokenu (slovo, číslo, ...) na řádek přidávají další informace oddělené znakem tabulátoru. Při syntaktickém zpracování jsou na řádek přidány například základní tvary slov, převedená slova na malá písmena, POS značky a informace pro sestavení syntaktického stromu.

Po syntaktickém zpracování následuje sémantické obohacení textu. Tato fáze je pro práci nejpodstatnější, jelikož dochází k rozpoznání pojmenovaných entit. K entitám jsou ještě přidány další informace ze znalostní báze, jako například odkaz na stránky Wikipedie. Jedním z možných výstupních formátu této komponenty je MG4J, který je vhodný pro indexaci stejnojmenným nástrojem [9].

4.4 Modul pro klasifikaci vztahu

Jádrem tohoto modulu je implementovaný model pro klasifikaci vztahu, jehož architektura je založena na předtrénovaném modelu sítě typu *transformers* pro kontextovou reprezentaci slov ve větě. Model na vstupu obdrží všechny výstupy z předchozího zpracování, přičemž je pro něho důležitá sekvence tokenů a rozpoznané entity. Z množiny pojmenovaných entit se vytvoří všechny možné kombinace dvojic entit ve větě a do sekvence tokenů se doplní tokeny pro ohraničení entit. Tato sekvence tokenů musí být přeložena na sekvenci tokenů podporované použitou sítí typu *transformers*, rozšířena o všechny potřebné speciální tokeny a zkrácena na maximální podporovanou vzdálenost. Z takto předzpracovaného vstupu se klasifikují vztahy mezi všemi dvojicemi entit. Výstupem se stává vstupní struktura obohacená o seznam trojic, kde první dvě položky udávají index do seznamu entit a třetí klasifikovaný vztah.

V následující kapitole bude nejdříve popsáno několik různých předtrénovaný modelů sítě typu *transformers* a následně navržený model pro klasifikaci vztahů mezi entitami.

4.4.1 Předtrénovaný model sítě typu transformers

Jak bylo popsáno v kapitole 3.2.2, aktuální řešení pro klasifikaci vztahů jsou založena na rekurentních neuronových sítích, konvolučních neuronových sítích, nebo sítích typu *transformers*. Mnoho aktuálně nejlepších řešení napříč různými úlohami z oblasti zpracování přirozeného jazyka využívá předtrénovaný model sítě typu *transformers*. To bylo důvodem pro použití sítě typu *transformers* v rámci návrhu modelu.

Nejznámější sítí je model BERT, který má v dnešní době několik veřejně dostupných předtrénovaných modelů. Reakcí na velmi dobré výsledky modelu BERT vzniklo několik

[7] Textový formát pro strukturované uložení dokumentu, podobný formátu XML.

jeho mutací. V této kapitole budou popsány dvě z nich, a to modely RoBERTa a ALBERT. Dalším zástupcem sítí typu *transformers*, popsaných v této kapitole, bude model XLNet. Autoři zmíněných modelů se navzájem ovlivňují, porovnávají své výsledky a zveřejňují další nové předtrénované modely. Proto nelze jednoznačně říci, která z architektur dosahuje lepších výsledků. Vždy záleží na konkrétní aplikaci a použitém předtrénovaném modelu.

Model v této práci je vystavěn na předtrénovaném modelu BERT, přičemž lze jednoduše při učení zvolit libovolný z ostatních zmíněnými předtrénovaných modelů. Využitím různých modelů se blíže zabývá kapitola 5.3.3 při zhodnocení systému.

BERT

Napříč různými úlohami sklidil velký úspěch předtrénovaný model BERT (*Bidirectional Encoder Representations from Transformers*) [12]. Jedná se o obousměrný kodér ze sítě typu *transformers* se stejnou architekturou, která byla popsaná v kapitole 2.2.3. Existují dva základní typy modelu: menší model $\text{BERT}_{\text{base}}$ s 12 vrstvami, 12 hlavami a dimenzí modelu $d_m = 768$, a vetší $\text{BERT}_{\text{large}}$ s 24 vrstvami, 16 hlavami a dimenzí modelu $d_m = 1024$. Model poskytuje vektorovou reprezentaci jednoduché věty, páru vět, ale i sekvenční vektorovou reprezentaci jednotlivých slov. Veřejně je dostupných několik předtrénovaných modelů[8].

Vstupní věta je převedena na posloupnost tokenů ze slovníku, rozšířena o počáteční token `[CLS]` a oddělovací token `[SEP]`. Sítě typu *transformers* pracují s pevnou šířkou vstupu, proto doplníme sekvenci do maximální délky speciálním tokenem `[PAD]`:

`[CLS] Věta A [SEP] [PAD]`* .

Pozice prvního tokenu výstupní vrstvy, tzn. token `[CLS]`, reprezentuje vektorovou reprezentaci celé věty. V případě úloh nad páry vět slouží token `[SEP]` také pro jejich oddělení:

`[CLS] Věta A [SEP] Věta B [SEP] [PAD]`* .

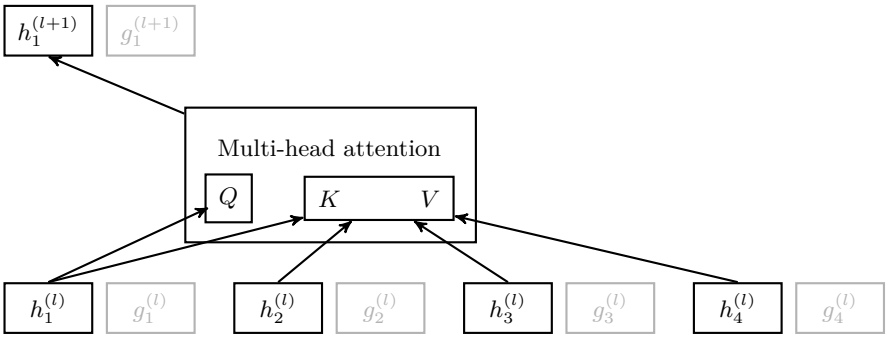
Model BERT byl trénován na dvou úlohách z oblasti zpracovávání přirozeného jazyka: predikce maskovaných slov (angl. *masked language modeling*) a predikce následujících vět (angl. *next sentence prediction*). U první úlohy se na vstupu náhodně zamění některá slova za speciální token `[MASK]` a následně se je model snaží správně predikovat. Tímto způsobem je model BERT učen jako autoenkodér, který se snaží opravit poškozený vstup na základě zbylé věty. Výhodou této metody je schopnost využití jak levého, tak pravého kontextu ve větě.

Při druhém úkolu má model BERT na vstupu dvě sekvence textu a má rozhodnout, zda druhá sekvence v pořadí následuje v dokumentu hned za první. Negativní případy jsou vytvořeny vybráním dvou sekvencí z dvou různých dokumentů. Trénování probíhalo na velkých datových sadách *BooksCorpus* [51] a anglické Wikipedie.

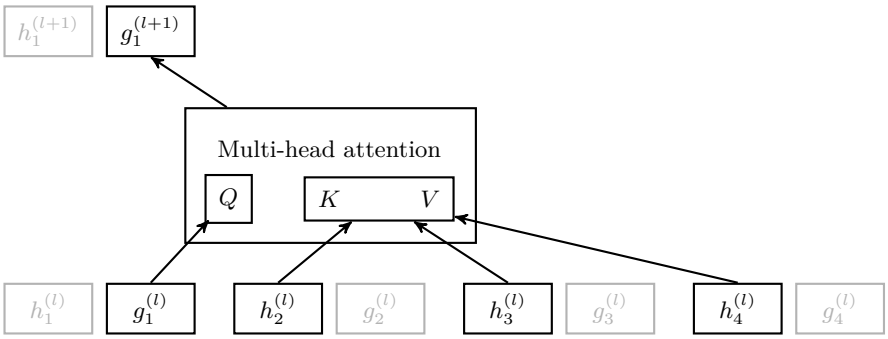
XLNet

Model XLNet [45] je dalším zástupcem sítí typu *transformers* podobný modelu BERT, ale již s výraznějšími odchylkami v architektuře. Modely založené na rekurentních sítích, nebo sítě typu *transformers* jako GPT, lze zařadit mezi autoregresní modely, jelikož využívají pro predikci následujícího slova levý nebo pravý kontext. Naproti tomu model BERT se řadí mezi auto-enkodéry, jelikož byl trénován na úloze, kterou lze popsat jako opravu vstupních

[8] https://github.com/google-research/bert



(a) Funkce pozornosti *content-stream attention*, která je stejná jako běžná funkce *self-attention*.



(b) Funkce pozornosti *query-stream attention*, pro kterou je skryta vektorová reprezentace aktuálního slova.

Obrázek 4.2: Blokové znázornění jedné vrstvy funkce pozornosti *Two-stream self-attention*. Obrázky pro přehlednost popisují výpočet kontextové reprezentaci jednoho slova z textu.

poškozených dat na základě celého vstupu. To znamená, že model BERT dokáže využít jak pravý, tak levý kontext. Model XLNet kombinuje oba přístupy, a to použitím permutací během trénování.

XLNet se při učení snaží predikovat postupně všechna slova ve větě na základě předchozího kontextu, na rozdíl od modelu BERT, který predikuje pouze maskovaná slova. Pokud by kontextem byla pouze předcházející slova ve větě, tak by ztratil schopnost využívat pravý kontext, což model BERT dokáže. Zde se dostávají ke slovu permutace množiny indexů jednotlivých slov ve větě. Tyto permutace udávají, v jakém pořadí se mají predikovat jednotlivá slova, přičemž kontextem se stávají všechna již predikovaná slova. Díky tomu je dosaženo výhod jak autoregresních modelů, tak možnosti využití pravého kontextu ve větě.

Dalším rozdílem oproti ostatním modelům je použití funkce pozornosti *two-stream self-attention*. Tato funkce se skládá ze dvou paralelních funkcí pozornosti pojmenovaných *content-stream attention* (obrázek 4.2a) a *query-stream attention* (obrázek 4.2b), přičemž funkce *content-stream attention* je totožná s běžnou funkcí *self-attention*. Pro funkci *query-stream attention* je skryta vektorová reprezentace aktuálně překládaného slova, přičemž dotazem se stávají vektory $g_i^{(l)}$ a výstupem jsou vektory $g_i^{(l+1)}$. Vektory jsou pro všechna slova na vstupu $g_i^{(0)}$ inicializovány na vektor w, jehož nalezení je předmětem učení. Výstupní kontextovou reprezentací se potom stává výstup poslední vrstvy N funkce *query-stream attention*, tedy vektory $g_i^{(N)}$.

Model XLNet využívá rekurentní mechanizmus Transformers-XL, který je schopný zpracovávat i věty delší, než je maximální délka vstupního segmentu modelu. Kvůli tomu pou-

žívá rozdílný formát vstupní věty, kde <sep> je oddělovací token, <cls> token reprezentující kontext celé věty a token <pad> slouží k vyplnění věty do maximální délky:

<pad>* Věta A <sep> <cls> .

Pro úlohy s dvojicí vět je formát obdobný:

<pad>* Věta A <sep> Věta B <sep> <cls> .

Přetrénované modely XLNet byly trénované na výrazně větších datech než model BERT. Také se lišily ostatní hyperparametry, zvláště velikost dávky o 8 tisících větách místo 256 [45]. Dostupné jsou opět dva typy modelů $\text{XLNet}_{\text{base}}$ a $\text{XLNet}_{\text{large}}$ se stejným počtem vrstev, hlav a dimenzí modelu d_m jako u modelu BERT.

RoBERTa

Model RoBERTa (*Robustly optimized BERT approach*) [27] vychází, jak je patrné z názvu, z architektury modelu BERT. Předtrénované modely RoBERTa se od modelu BERT liší zvláště v reprezentaci vstupního textu, ve zvolených hyperparametrech učení a větší trénovací datové sadě.

Drobné nuance implementace se týkají převážně reprezentace vstupního textu. Model RoBERTa používá obdobné schéma vstupních dat jako model BERT, kde <s> je počáteční token, </s> oddělovací token a <p> je výplň:

<s> Věta A </s> <p>* .

Schéma pro dvojici vět se liší, kde pro oddělení jednotlivých vět je zvolena dvojice oddělovacích tokenů:

<s> Věta A </s></s> Věta B </s> <p>* .

Model využívá jinou implementaci vektorové reprezentace vstupních slov, která obsahuje 50 tisíc slov oproti 30 tisícům v modelu BERT. Pokud se některé slovo nenachází v slovníku, tak je rozděleno do několika řetězců a ty jsou následně převedeny na vektorovou reprezentaci. Díky tomu lze libovolné vstupní slovo převést na vektorovou reprezentaci a nemusí se využívat speciální token pro neznámá slova.

Model byl trénován pouze na úloze predikce slov, kdy se využívá dynamické místo statického maskování jednotlivých slov. Model BERT provede maskování vstupních slov při předzpracování, a v každé epoše trénování se tedy predikují stejná zamaskovaná slova. V modelu RoBERTa se každý vzorek při předzpracování vstupních dat duplikuje desetkrát s různými maskovanými slovy. Ve fázi učení se v každé epoše využije pouze jedna instance ke každému vzorku dat. Když je tedy model trénován v čtyřiceti epochách, tak je každá instance vzorku viděna právě čtyřikrát. Tento způsob autoři nazvali jako dynamické maskování. Při učení byla vstupní data rozšířena, a to desetkrát z 16 GB na 160 GB. Také byla zvětšena velikost dávky z 256 na 8 tisíc vět.

ALBERT

Model ALBERT (*A Lite BERT*) [23] je další mutací modelu BERT, která si klade za cíl snížení paměťových nároků a snížení počtu parametrů modelu optimalizovaných během učení. Nevýhodou těchto úprav je menší úspěšnost při porovnání s ostatními modely se stejnými parametry, která je ale kompenzována možností trénování výrazně větších modelů

Model	Parametrů	Vrstev	Hlav	d_m	d_e	Sdílené par.
$\text{BERT}_{\text{base}}$	108 M	12	12	768	768	Ne
$\text{BERT}_{\text{large}}$	334 M	24	16	1024	1024	Ne
$\text{ALBERT}_{\text{base}}$	12 M	12	12	768	128	Ano
$\text{ALBERT}_{\text{large}}$	18 M	24	16	1024	128	Ano
$\text{ALBERT}_{\text{xlarge}}$	60 M	24	32	2048	128	Ano
$\text{ALBERT}_{\text{xxlarge}}$	235 M	12	64	4096	128	Ano

Tabulka 4.1: Seznam druhů předtrénovaných modelů ALBERT [23]. Parametr d_e udává velikost dimenze vstupní bezkontextové reprezentace slov.

za kratší dobu. Seznam typů předtrénovaných modelů, včetně srovnání s modely sítě BERT, se nachází v tabulce 4.1.

Odlišnost od modelu BERT spočívá v redukci dimenze vstupní nekontextové reprezentace slov, sdílením parametrů napříč jednotlivými vrstvami a využitím predikce pořadí vět (angl. *sentence-order prediction*), místo predikce následující věty, při trénování modelu. První dvě odlišnosti vedou právě k snížení paměťové náročnosti a počtu parametrů k učení.

U všech typů modelu ALBERT je zvolena dimenze vektorové reprezentace $d_e = 128$, kdežto u modelů BERT a RoBERTa je dimenze $d_e = d_m$. Snížení dimenze vektorové reprezentace, která je až následně transformována do dimenze d_m, vede k menší velikosti slovníku a snížení počtu parametrů k optimalizaci.

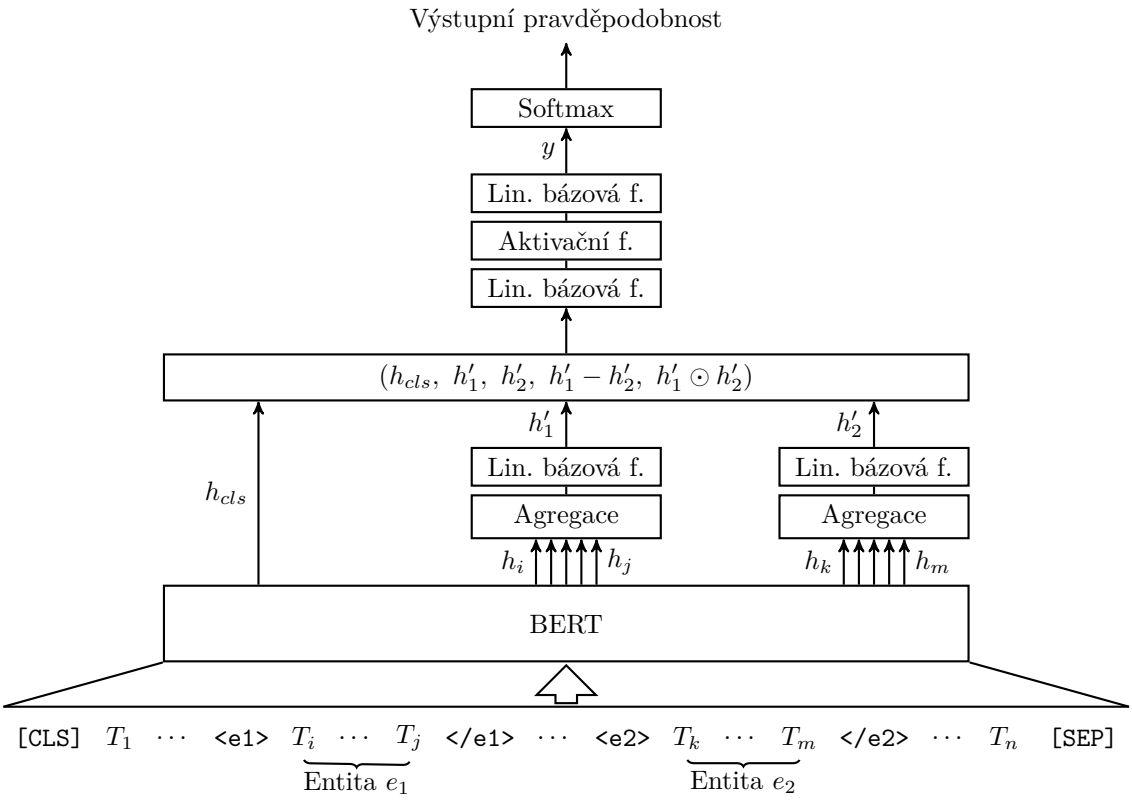
Jednotlivé vrstvy modelu sdílejí spolu všechny parametry. Tím se výrazně sníží počet potřebných parametrů potřebných k trénování a jejich počet je zvláště závislý na dimenzi d_m (vizte tabulku 4.1). Pro srovnání, počet potřebných parametrů k optimalizaci je v modelu $\text{ALBERT}_{\text{large}}$ 18-krát menší než v modelu $\text{BERT}_{\text{large}}$.

Úloha predikce pořadí vět definovaná autory modelu pro zefektivnění učení je velmi podobná úloze predikce následující věty z modelu BERT, jen se liší ve způsobu vytvoření trénovací datové sady. V obou případech se pozitivní případy vytvoří extrakcí dvou po sobě jdoucích bloků textu v jednom dokumentu. Při predikci následující věty se negativní vzorky vytvoří jako dvojice bloků textu ze dvou různých dokumentů. Negativní vzorky při úloze predikce pořadí vět se vytvoří pouze záměnou pořadí bloků z pozitivních případů.

4.4.2 Model pro klasifikaci vztahů mezi pojmenovanými entitami

Model pro klasifikaci vztahů mezi pojmenovanými entitami byl vytvořen pomocí metody *dotrénování* předtrénovaného modelu BERT. V rámci vyhodnocení systému bylo vyzkoušeno také několik dalších modelů popsaných v předchozí kapitole. Při použití této metody postačí pouze několik málo epoch učení, jelikož předtrénovaný model již poskytuje mnoho potřebných informací a učí se hlavně přidané poslední vrstvy.

Jako vstup obdrží model větu obsahující dvě entity e_1 a e_2. Nezávisle na sobě se ukázalo jako výhodné ohraničit jednotlivé entity v textu speciálními symboly, a dodat tak modelu dodatečné informace o pozici entit [5, 44]. Pro ohraničení entity e_1 se využijí dva speciální tokeny `<e1>` a `</e1>`, analogicky pro druhou entitu e_2 tokeny `<e2>` a `</e2>`. Některé předtrénované modely BERT mají za tímhle účelem rezervované nevyužité tokeny ve slovníku, bohužel modely ostatních architektur nevyužité tokeny neobsahují. Proto je slovník rozšířen o čtyři nové tokeny, které jsou inicializovány na náhodnou hodnotu.



Obrázek 4.3: Architektura modelu *RelC-Transformers* pro klasifikaci vztahu mezi pojmenovanými entitami.

Vstupní věta je také rozšířena o token `<cls>`, reprezentující vektorovou reprezentaci celé věty, a ukončovací token `<eos>`. Jak bylo v předchozí kapitole popsáno, jednotlivé modely využívají trochu odlišný vstupní formát, zvláště s různou pozicí tokenu `<cls>`. Při tokenizaci textu je použit správný formát dle zvoleného předtrénovaného modelu. Pro jednoduchost se dále v kapitole bude uvažovat formát modelu BERT, kde je token `<cls>`=`[CLS]` a token `<eos>`=`[SEP]`.

Návrh modelu je znázorněn na obrázku 4.3. Předtrénovaný model kodéru poskytne vektorovou reprezentaci každého vstupního tokenu h_t a vektorovou reprezentaci celé věty h_{cls}, neboli vektorovou reprezentaci tokenu `<cls>`. Pro klasifikaci vztahu mezi entitami by postačil pouze vektor h_{cls}, ale pro zlepšení výsledků klasifikace se vytvoří navíc reprezentace dané dvojice entit.

Vektory h_i a h_j odpovídají prvnímu a poslednímu tokenu entity e_1 a obdobně vektory h_m a h_k prvnímu a poslednímu tokenu entity e_2. Vektorová reprezentace jednotlivých entit se vytvoří pomocí agregace vektorů dané entity. Pro entitu e_1 tedy vytvoříme vektorovou reprezentaci h'_1 jako průměr vektorů h_i až h_j a pro entitu e_2 vektorovou reprezentaci y'_2 z vektorů h_m až h_k. Následně se nad zprůměrovanými vektory aplikuje plně propojená dopředná vrstva. Formálně lze vyjádřit vektorovou reprezentaci entity e_1 jako vztah (4.1) a reprezentaci entity e_2 jako vztah (4.2).

$$h'_1 = W_1 \cdot \frac{\sum_{t=i}^{j} h_t}{j - i + 1} + b_1 \tag{4.1}$$

$$h'_2 = W_2 \cdot \frac{\sum_{t=k}^{m} h_t}{m-k+1} + b_2 \tag{4.2}$$

Všechny matice vah jsou stejné dimenze $W_0, W_1, W_2 \in \mathbb{R}^{d_m \times d_m}$, stejně tak $b_0, b_1, b_2 \in \mathbb{R}^{d_m}$. Kapitola 5 se mimo jiné zabývá využitím jiných agregačních funkcí, jako použitím prvního pozičního tokenu entity (pro entitu e_1 vektor h_{i-1} a pro entitu e_2 vektor h_{k-1}) [5], nebo funkce *max-pooling* z konvolučních sítí.

Reprezentace dvojice pojmenovaných entit se vytvoří za použití tří operátorů: zřetězení vektorové reprezentace entit concat(h'_1, h'_2), rozdílu vektorů $h'_1 - h'_2$ a Hadamardova součinu[9] vektorů $h'_1 \odot h'_2$. Operátor zřetězení pro vytvoření reprezentace dvojice entit je využit v několika pracích [5, 44, 49]. Zbylé dva operátory dodávají navíc další užitečné informace.

Reprezentace celé věty h_{cls} se zřetězí s reprezentací dvojice entit a přidá se dopředná neuronová síť FNN.

$$y = \text{FFN}\left(\text{concat}\left(h_{cls}, h'_1, h'_2, h'_1 - h'_2, h'_1 \odot h'_2\right)\right) \tag{4.3}$$

Samotná dopředná síť FFN se skládá ze dvou vrstev, kde σ je nelineární aktivační funkce.

$$\text{FFN}(x) = W_4 \cdot \sigma\left(W_3 \cdot x + b_3\right) + b_4 \tag{4.4}$$

Matice vah $W_3 \in \mathbb{R}^{5 \cdot d_m \times d_h}$, $W_4 \in \mathbb{R}^{d_h \times L}$ a $b_3 \in \mathbb{R}^{d_h}$, $b_4 \in \mathbb{R}^{L}$, kde d_h je hyperparametr a L je počet tříd, do kterých se klasifikuje vztah mezi pojmenovanými entitami. Jako aktivační funkce σ byl zvolen hyperbolický tangens (tanh).

Poslední vrstvou je vrstva *softmax* a pravděpodobnost vztahu r mezi entitami lze vyjádřit pomocí vztahu (4.5).

$$p(r|y) = \frac{e^{y_r}}{\sum_{n=1}^{L} e^{y_n}} \tag{4.5}$$

Před každou plně propojenou vrstvou je v průběhu učení aplikován mechanismus *dropout*[10].

Model je implementován v jazyce Python 3 s použitím knihovny *transformers*[11] pro předtrénovaný model a knihovny *pytorch*[12] pro přidání jednotlivých vrstev popsaných v této kapitole. Pokud se v počítači nachází GPU, tak je požito pro akceleraci učení a klasifikace.

4.5 Následné zpracování a uložení

Posledním modulem v řadě je modul pro uložení výsledků analýzy. Modul na vstupu obdrží všechny výsledky jednotlivých částí zpracování a podle zvolené implementace provede jejich uložení. Na rozdíl od modulů pro extrakci vět ze vstupních souborů lze ukládat pouze do jednoho formátu.

V rámci práce jsou výsledky uloženy ve formátu N-Triples. Tento formát je podmnožinou komplexnějšího formátu Turtle a slouží k serializaci RDF závislostních grafů, nebo znalostních bází jako např. Wikidata. Data jsou uložena formou prostého textu, kdy je na každém řádku uložena jedna hrana grafu, nebo v našem případě jeden vztah. Každý řádek je složen ze čtyř částí: podmětu, predikátu, předmětu a tečky, které jsou odděleny libovolným bílým znakem:

[9] Hadamardův součin (angl. *Hadamard product*, nebo *element-wise product*) je součin vektorů, resp. matic, po jednotlivých složkách.

[10] *Dropout* je regulační metoda pro zmírnění přetrénování (angl. *overfitting*) neuronových sítí.

[11] https://github.com/huggingface/transformers

[12] https://pytorch.org/

<podmět> <predikát> <předmět> .

Modul na vstupu obdrží množinu pojmenovaných entit a vztahy mezi nimi. Před uložením dojde k filtraci speciálního vztahu ∅ (žádný, nebo neznámý vztah). Vztah je do formátu N-Triples vložen na pozici predikátu a entity na pozice podmětu s předmětem tak, aby byl zachován správný směr vztahu. Formát entity v souboru je následující:

<kategorie>:"<entita>".

Intuitivně term <kategorie> reprezentuje typ rozpoznané pojmenované entity a term <entita> extrahovanou posloupnost slov.

4.6 Shrnutí

V kapitole byl popsán návrh systému pro extrakci vztahů mezi pojmenovanými entitami v textu na webu. Systém se skládá ze čtyř podstatných částí: zpracování staženého obsahu internetu, rozpoznání pojmenovaných entit v textu, klasifikace vztahu mezi entitami a uložení výsledků do souboru. Celý systém je modulární, proto si lze jednotlivé části za pomoci konfiguračního souboru přizpůsobit nebo nahradit podle konkrétní aplikace.

Jako základní podporované formáty vstupních souborů byly implementovány: prostý text, HTML, WARC a MG4J. Pro syntaktickou analýzu a rozpoznání pojmenovaných entit byly využity nástroje *SpaCy* a *Corpproc*. Součástí kapitoly je také krátká rešerše těchto systémů. Samotný model pro klasifikaci vztahu mezi entitami je založen na veřejně dostupných předtrénovaných kodérech ze sítě typu *transformers*. Kapitola obsahuje rešerši modelů BERT, XLNet, RoBERTa a ALBERT, jejichž využitím se blíže zabývá následující kapitola. Výsledky extrakce vztahů jsou uloženy ve formátu N-Triples.

Kapitola 5

Zhodnocení

Následující kapitola se bude zabývat vyhodnocením modelu pro klasifikaci vztahů mezi pojmenovanými entitami, který je jádrem implementovaného systému. Před samotnými experimenty budou popsány metriky pro jejich vyhodnocení a datové sady. Pro vyhodnocení byly využity běžně rozšířené metriky z oblasti zpracování přirozeného jazyka a klasifikace. Experimenty byly provedeny na trojici různých datových sad se zaměřením na nuance v architektuře systému a použití různých předtrénovaných modelů.

5.1 Metriky

Při vyhodnocení klasifikace nás zajímají čtyři statistické údaje. Přirozeně nás zajímá počet správně klasifikovaných objektů t_p a počet správně *neklasifikovaných* objektů t_N do dané třídy. Analogicky k tomu značíme počet chybně klasifikovaných objektů f_p a počet chybně *neklasifikovaných* objektů do dané třídy f_n.

Pro hodnocení úspěšnosti nástrojů pro analýzu přirozeného textu se využívají metriky přesnost (angl. *precision*) a výtěžnost (angl. *recall*), které jsou založené na předchozích statistikách. Přesnost, definovaná vztahem (5.1), je míra správně klasifikovaných objektů ke všem výsledkům analýzy.

$$\text{precision} = \frac{t_p}{t_p + f_p} \tag{5.1}$$

Výtěžnost je míra správně klasifikovaných objektů k celkovému počtu objektů, které měly být do dané třídy klasifikovány (vizte vztah (5.2)).

$$\text{recall} = \frac{t_p}{t_p + f_n} \tag{5.2}$$

Další využívanou metrikou v statické analýze je F_β, definovaná vztahem (5.3), kdy se volí obvykle $\beta = 1$.

$$F_\beta = \left(1 + \beta^2\right) \cdot \frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}} \tag{5.3}$$

Metrika F_1 je harmonickým průměrem mezi přesností a výtěžností. Když dosahuje metrika F_1 hodnoty blízké 1, tak se dosahuje velmi dobré přesnosti i výtěžnosti.

5.1.1 Klasifikace do více tříd

Předchozí metriky se počítají pro každou třídu zvlášť, proto je potřeba zavést pro klasifikaci do tří a více tříd způsob agregace výsledků. Běžně se používá makro průměr, vážený průměr a mikro průměr.

Při makro průměru se vypočítají jednotlivé metriky pro každou třídu zvlášť a následně se provede aritmetický průměr. Jedná se o nejjednodušší přístup a lze jej využít při vyvážených datech, jelikož každá třída má stejný vliv na výsledek bez ohledu na počet vzorků třídy.

Vážený průměr je podobný makro průměru, jen se místo aritmetického průměru využije vážený. Vážený průměr lze využít tam, kde některé třídy mají větší význam než jiné. Také se může zvýšit vliv tříd s více vzorky u nevyvážených dat.

U mikro průměru se nejdříve vypočítá celkový počet správně klasifikovaných vzorků t_p a chybně klasifikovaných f_p a f_n.

$$\text{micro-precision} = \frac{\sum_{i=1}^{N} t_p^i}{\sum_{i=1}^{N} t_p^i + f_p^i} \tag{5.4}$$

$$\text{micro-recall} = \frac{\sum_{i=1}^{N} t_p^i}{\sum_{i=1}^{N} t_p^i + f_n^i} \tag{5.5}$$

$$\text{micro-}F_1 = 2 \cdot \frac{\text{micro-precision} \cdot \text{micro-recall}}{\text{micro-precision} + \text{micro-recall}} \tag{5.6}$$

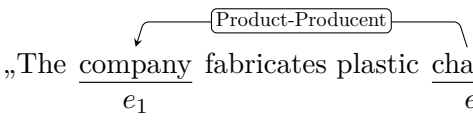
Následně se z nich postupně vypočítá přesnost, výtěžnost a F_1 (vizte rovnice výše). Mikro průměr nejlépe odráží celkovou úspěšnost klasifikace při výrazně nevyvážených datech.

5.2 Datové sady

Pro porovnání a experimentování s modely pro extrakci vztahů z textu bylo v minulosti vytvořeno několik datových sad. Pro vyhodnocení byly zvoleny tři datové sady.

5.2.1 SemEval-2010 Task 8

Datová sada *SemEval-2010 Task 8* [15] se skládá z 10717 anglických vět obsahujících právě dvě anotované entity, z čehož 8000 příkladů slouží k trénování modelu a 2717 k jeho validaci. Datová sada obsahuje devět sémantických vztahů: *Cause-Effect*, *Instrument-Agency*, *Product-Producer*, *Content-Container*, *Entity-Origin*, *Entity-Destination*, *Component-Whole*, *Member-Collection*, *Message-Topic* a speciální vztah *Other*, pokud mezi entitami neexistuje vztah nebo se jedná o jiný neznámý vztah. Ukázkou z datové sady může být například sémantický vztah *Product-Producent* obsažený v anotované větě



U každého z devíti vztahů záleží na správném pořadí entit, takže ve výsledku se klasifikuje do 19 tříd. Datová sada *SemEval-2010 Task 8* je hojně využívána k porovnání různých architektur modelů, a proto je její součástí i hodnotící skript, který počítá makro-průměrovanou F_1 z 9 vztahů bez vztahu *Other*, kdy je brán v potaz správný směr vztahu mezi entitami. Přehled nejúspěšnějších řešení[1] se nachází v tabulce 5.1.

[1] https://paperswithcode.com/sota/relation-extraction-on-semeval-2010-task-8 [cit. 6. 5. 2020].

Model	Metoda	F_1 [%]	Rok
EPGNN [49]	$\text{BERT}_{\text{base}}$ + GNN	90,20	2019
BERTEM+MTB [5]	$\text{BERT}_{\text{large}}$	89,50	2019
R-BERT [44]	$\text{BERT}_{\text{base}}$	89,25	2019
KnowBert-W+W [34]	$\text{BERT}_{\text{base}}$ + knowledge base	89,10	2019
Entity-Aware BERT [42]	BERT_{SP} + entity-aware	89,00	2019
Att-Pooling-CNN [43]	CNN	88,00	2016
TRE [2]	GPT	87,10	2019
Entity Attention Bi-LSTM [24]	Bi-LSTM + self-attention	85,20	2019
REDN [26]	$\text{BERT}_{\text{base}}$	*91,00	2020

Tabulka 5.1: Žebříček zveřejněných výsledků modelů, které byly evaluovány na datové sadě *SemEval-2010 Task 8* ke dni 6. 5. 2020. U modelu REDN nepoužili autoři přiložený hodnotící skript, ale micro-F_1, proto nelze porovnat s ostatními řešeními.

5.2.2 KBP37

Datová sada *KBP37* [47] je složena z 21046 anotovaných anglických vět. Pro trénovaní je využito 15917 vět, pro validaci 3405 a pro vývoj 1724. Datová sada obsahuje 18 vztahů a speciální vztah *no_relation*, pokud mezi entitami není žádný známý vztah (vizte přílohu B.1). U každého ze vztahů záleží na pořadí entit v textu, proto se ve výsledku klasifikuje do 37 tříd. Datová sada sdílí stejný formát s předchozí datovou sadou *SemEval-2010*. Jako příklad lze uvést větu obsahující vztah *per:countries_of_residence*

per:countries_of_residence

„Mohammad Musa Shafiq was Prime Minister of Afghanistan.“

e_1 e_2

Na rozdíl od datové sady *SemEval* jsou dvojice slov ve vztahu vždy pojmenované entity. Pojmenované entity se často skládají z více jak jednoho slova, což je v předchozí datové sadě ojedinělé. Průměrná délka vět je v datové sadě *KBP37* také výrazně větší. Mezi anotovanými entitami může být více sémantických vztahů, i když je v datové sadě uveden právě jeden vztah jako odpověď. Tyto případy by se měly vyskytovat pouze vzácně.

5.2.3 FewRel

Další použitá datová sada při vyhodnocení systému se nazývá *FewRel* (*Few-Shot Relation Classification Dataset*) [14]. Tato datová sada není určena k tradiční klasifikaci vztahů, ale pro problém, kdy učíme model na jiné množině vztahů, než do kterých následně klasifikujeme. Datová sada vznikla ze zpracovaného obsahu Wikipedie, kde byly rozpoznány pojmenované entity a jim přiřazena položka ze znalostní báze Wikidata. Z takto zpracovaného obsahu byly vybrány věty obsahující dvě entity, mezi kterými existuje vztah ve znalostní bázi, a následně prošly ruční kontrolou profesionálními anotátory. Datová sada obsahuje 70000 anotovaných vět a 100 různých vztahů ze znalostní báze Wikidata, kde každá třída obsahuje právě 700 vzorků. Datová sada byla rozdělena do 3 částí, přičemž trénovací sada obsahuje 64, vývojová sada 16 a testovací sada zbylých 20 vztahů. Testovací sada není zveřejněna a slouží k objektivnímu vyhodnocení různých systémů[2]. Seznam všech

[2] http://www.zhuhao.me/fewrel/

Datová sada	Počet vzorků	Počet vztahů	Průměrná délka	Průměrná vzdálenost
SemEval-2010 Task 8	10717	9+1	18,35	3,62
KBP37	21046	18+1	31,11	7,39
FewRel	70000	100	24,99	-
RCD-FewRel	63725	80+1	26,77	7,17

Tabulka 5.2: Srovnání jednotlivých datových sad. U datové sady *FewRel* není zveřejněna testovací část, proto jsou uvedeny pouze zveřejněné údaje z článku [14].

vztahů se nachází v příloze B.2. Vztahy pokrývají rozmanitou škálu témat jako jsou např. sport, umění, politika, geografie nebo rodinné vazby.

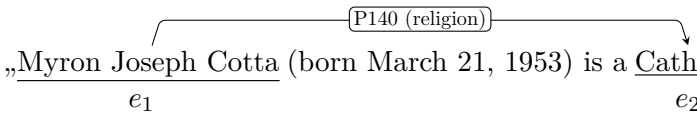
Statistika

Předchozí části kapitoly obsahovaly představení tří použitých datových sad pro vyhodnocení systému. Při srovnání datových sad (vizte tabulku 5.2) datová sada *SemEval* obsahuje menší počet obecnějších vztahů. Vztahy se také klasifikují mezi pojmy, které v mnoha případech nejsou pojmenované entity a jen zřídka se skládají z více jak jednoho slova. Datová sada se stala oblíbenou pro srovnání různých modelů, a za tímto účelem je použita i v této práci.

V reakci na tyto charakteristiky vznikla datová sada *KBP37*, která obsahuje dvojnásobný počet vztahů mezi víceslovnými pojmenovanými entitami. Navíc obsahuje výrazně delší věty, které se v průměru skládají asi z 31 slov, resp. tokenů. Také pojmenované entity jsou průměrně od sebe více jak dvakrát vzdálenější. Datová sada je ve vyhodnocení použita pro srovnání, jak jsou jednotlivé předtrénované modely úspěšné při klasifikaci vztahů v dlouhých větách se vzdálenými entitami.

Poslední datovou sadou je *FewRel*, která je ze všech největší, ať počtem vzorků, nebo počtem vztahů. Vztahy i pojmenované entity jsou propojeny se znalostní bází Wikidata. Délka vět je v datové sadě větší než v datové sadě *SemEval*, ale ne tak dlouhá jako v *KBP37*. Díky svému velkému počtu vztahů se tato datová sada stala základem pro model použitý v implementovaném systému. Pro lepší pochopení datové sady byla zhotovena statistika nejčastějších slov a spoluvýskytu dvojic slov ve stejné větě. Před vytvořením statistiky byla z vět nejdříve odstraněna nejčastější slova v angličtině, tzv. *stop words*, interpunkce a ostatní nealfanumerické znaky. Výsledky pro jednotlivé vztahy se nacházejí v příloze B.2.

Nejčastějším slovem napříč jednotlivými vztahy je slovo *born*. Typickým případem, obsahujícím toto slovo, může být následující věta:



Z toho lze usuzovat, že podstatná část vzorků se skládá z první věty článků na Wikipedii. Tyto věty jsou často velmi podobné a doménově závislé. Model trénovaný na této datové sadě může tedy dosahovat horších výsledků na vstupních datech z jiných zdrojů, jako např. novinových článků. Tento problém může být námětem pro budoucí práci.

Statistika počtu a spoluvýskytu slov také ukázala podobnost v kontextu vět obsahujících některé ze vztahů. Nejvýrazněji tomu je u případů z oblasti rodinných vazeb (vizte tabulku v příloze B.2, vztahy P22, P25, P40, ...).

Datová sada RCD-FewRel

Na základě datové sady *FewRel* byla pro potřeby této práce vytvořena datová sada pro klasifikaci vztahů mezi pojmenovanými entitami *RCD-FewRel* (*Relation Classification Dataset based on FewRel*). Jako základ datové sady byla použita trénovací a vývojová část datové sady *FewRel*. Tím se získaly pozitivní případy u 80 různých vztahů o 700 vzorcích, tedy dohromady 56000 pozitivních vzorků.

Negativní vzorky se získaly z archívu Wikipedie, která byla zpracována nástroji projektu *Corpproc*. Nástroje při zpracování Wikipedie nabízejí kromě samotných referencí z Wikipedie na ostatní články také rozpoznané pojmenované entity a jejich propojení se znalostní bází. Aby se v následujících krocích vytváření datové sady nezanesly další chyby způsobené automatizovaným rozpoznáním entit a jejich propojením se znalostní bází, byly jako pojmenované entity chápány pouze reference na články projektu Wikipedie.

Ze zpracovaných souborů MG4J byly vyfiltrovány pouze věty obsahující dvě a více pojmenovaných entit a odkazy byly převedeny na položku v znalostní bázi Wikidata. Pro zaindexování znalostní báze a následné dotazování byl využit nástroj *QLever* [7]. Pomocí znalostní báze se určily všechny sémantické vztahy mezi entitami. Tím byly vytvořeny základy pro získání negativních případů.

Každé větě se následně určila množina kandidátů na vztah mezi entitami. Vztah je považován za kandidáta, pokud věta obsahuje některou z pěti nejčastějších dvojic slov, které obsahují pozitivní případy pro daný vztah. Hlavní myšlenkou tohoto kroku je, že věta obsahující např. slova *prime* a *minister* bude pravděpodobněji obsahovat vztah P6 (představitel, angl. *head of government*), než jiné věty. Pro každý vztah bylo náhodně vybráno maximálně 100 vět, obsahujících daný vztah v množině kandidátů, a dvě entity nacházející se ve větě, mezi kterými neexistoval v znalostní bázi žádný vztah. Negativní případy lze rozdělit do následujících tří kategorií:

1. Ve větě neexistuje vztah mezi pojmenovanými entitami, tedy ani mezi entitami, pro které ho chceme klasifikovat.

2. Ve větě existuje některý hledaný vztah mezi pojmenovanými entitami, ale ne mezi entitami, pro které ho chceme klasifikovat.

3. Ve větě existuje některý hledaný vztah, a to právě mezi entitami, pro které ho chceme klasifikovat. Případy v této kategorii jsou chybné, jelikož se jedná o pozitivní případy, a jsou převážně způsobeny neúplností znalostní báze. Namátkovou kontrolou bylo ověřeno, že případy v této kategorii nejsou příliš časté.

Datová sada byla převedena do formátu datových sad *SemEval* a *KBP37*, kde každému vztahu byl určen směr a byla přidána speciální třída *no_relation* pro negativní případy. Ve výsledku se provádí klasifikace do 161 tříd, a tím se jedná o největší datovou sadu s největším počtem tříd. Datová sada je rozdělena do tří částí: trénovací (60 %), vývojová (20 %) a testovací část (20 %). Při rozdělení datové sady obsahuje každá část všech 80 vztahů a vzorky každé třídy jsou mezi nimi rovnoměrně rozděleny.

Předmětem dalších prací může být rozšíření této datové sady o další vztahy a vzorky, prozkoumání jiných sofistikovanějších přístupů pro vytvoření negativních případů, nebo jejich ruční anotace.

Parametr	BERT	XLNet	RoBERTa	ALBERT
Maximální délka věty	128	128	128	128
Počet epoch	5	5	6	5
Koeficient učení	$2 \cdot 10^{-5}$	$3 \cdot 10^{-5}$	$2 \cdot 10^{-5}$	$2 \cdot 10^{-5}$
Velikost dávky	16 (32)	16 (32)	16 (32)	16 (32)
Míra *dropout*	0,1	0,1	0,1	0,1

Tabulka 5.3: Zvolené hyperparametry modelu pro experimenty.

Reprezentace dvojice entit	SemEval	KBP37	RCD-FewRel
$\emptyset$	88,78	64,81	78,04
(h'_1, h'_2)	89,09	64,70	78,31
$(h'_1 - h'_2)$	89,21	64,94	78,27
$(h'_1 \odot h'_2)$	89,01	65,10	78,09
$(h'_1, h'_2, h'_1 - h'_2)$	89,10	64,90	78,12
$(h'_1, h'_2, h'_1 \odot h'_2)$	89,14	64,93	78,30
$(h'_1 - h'_2, h'_1 \odot h'_2)$	89,24	65,15	**78,35**
$(h'_1, h'_2, h'_1 - h'_2, h'_1 \odot h'_2)$	**89,25**	**65,28**	78,31

Tabulka 5.4: Výsledky experimentů s různými implementacemi reprezentace dvojice entit. Symbol $\emptyset$ znamená využití pouze kontextové reprezentace celé věty h_{cls}.

5.3 Experimenty

Experimenty se skládají ze 3 částí: hledání hyperparametrů, reprezentace dvojice entit a srovnání předtrénovaných modelů sítě typu *transformers*.

5.3.1 Parametry experimentů

Pro zhodnocení modelu pro klasifikaci vztahů mezi pojmenovanými entitami byly využity datové sady *SemEval Task 8*, *KBP37* a *RCD-FewRel*, které byly popsány dříve v této kapitole. Metrikou pro srovnání byla zvolena metrika F_1, konkrétně pro datovou sadu *SemEval* formou hodnotícího skriptu a pro zbylé dvě se jedná o metriku makro-F_1. Každý experiment byl vyhodnocen 10-krát a výsledkem je průměrná hodnota F_1.

Zvolené hyperparametry pro experimenty jsou uvedeny v tabulce 5.3. Jednotlivé parametry jsou stejné pro všechny tři datové sady, s výjimkou velikosti dávky, která byla pevně zvolena 16 pro datovou sadu *SemEval* a pro ostatní 32. Parametr počtu epoch e a koeficient učení η byl hledán v prostoru $e \in \{2, 3, 4, 5, 6, 7\}$ a $\eta \in \{1 \cdot 10^{-5}, 2 \cdot 10^{-5}, 3 \cdot 10^{-5}, 4 \cdot 10^{-5}, 5 \cdot 10^{-5}\}$.

5.3.2 Reprezentace dvojice entit

První sada experimentů je zaměřena na vyhodnocení vektorové reprezentace dvojice entit a použité agregační funkci. Jednotlivé experimenty jsou vyhodnocovány na vývojových částech datové sady *KBP37* a *RCD-FewRel* a testovací sadě *SemEval* kvůli absenci vývojové části.

Agregační funkce	SemEval	KBP37	RCD-FewRel
Repr. token	89,13	65,02	78,21
Průměr	89,25	**65,28**	**78,31**
Max-pool	**89,36**	65,02	77,98

Tabulka 5.5: Výsledky experimentů s různými funkcemi pro vytvoření reprezentace víceslovných entit. Agregační funkce „reprezentativní token“ představuje využití vektorové reprezentace první poziční značky entity, tzn. vektorovou reprezentaci tokenu `<e1>` a `<e2>`.

Model	SemEval	KBP37	RCD-FewRel
$\text{BERT}_{\text{base}}$ (uncased)	**89,25**	65,69	77,92
$\text{XLNet}_{\text{base}}$ (cased)	89,18	66,00	**78,18**
$\text{RoBERTa}_{\text{base}}$	88,71	**66,54**	77,46
$\text{ALBERT}_{\text{base}}$ (2. verze)	87,84	64,75	75,54

Tabulka 5.6: Výsledky srovnání naučených modelů s využitím různých předtrénovaných sítí pro kontextovou reprezentaci vstupní věty. Informace v závorkách identifikují konkrétní předtrénovaný model v knihovně *transformers*.

Jako reprezentace dvojice entit byly postupně zvoleny všechny kombinace operátorů, včetně využití pouze kontextové reprezentace celé věty (v tabulce 5.4 označeno znakem $\emptyset$). Využití pouze reprezentace věty již dává dobré výsledky a model je schopen klasifikovat vztah mezi entitami, ale přidáním reprezentace dvojice entit dochází k zlepšení klasifikace. Výjimkou je použití pouze operátoru zřetězení při datové sadě *KBP37*, kdy dojde k zhoršené klasifikaci, ale to lze vzhledem k ostatním výsledkům považovat za zašumělou hodnotu. Nejlepších výsledků se dosahuje při použití všech tří operátorů při datových sadách *SemEval* a *KBP37*, a použití operátorů rozdílu a Hadamardova součinu při datové sadě *RCD-FewRel*.

Další část experimentů je založena na porovnání jednotlivých agregačních funkcí a výsledky se nachází v tabulce 5.5. Jako agregační funkce lze využít reprezentační vektor (tedy vektor tokenu `<e1>` a `<e2>`), průměr vektorů slov entity nebo funkce max-pool. Jako nejlepší variantou podle zvolené metriky vychází použití průměrů vektorů. Sice u datové sady *SemEval* dosahuje nejlepšího výsledku funkce max-pool, ale jelikož sada obsahuje pouze malý počet případů, kdy datová sada obsahuje více jak jedno slovo, tak se průměr od max-pool téměř neliší.

5.3.3 Předtrénované modely sítě typu transformers

Model pro klasifikaci vztahů mezi entitami byl navržen a vybudován na předtrénovaném modelu BERT. Tato kapitola pojednává o využití jiných architektur síti typu transformers, a to konkrétně XLNet, RoBERTa a ALBERT. Pro srovnání byly využity modely typu *base*, jelikož oproti větším modelům mají výrazně nižší paměťové nároky, kratší dobu učení a rychlejší vyhodnocení. Ke každé architektuře byl vybrán některý z dostupných předtrénovaných modelů z knihovny *transformers*. Výsledky experimentů se nachází v tabulce 5.6. Výsledné hodnoty F_1 jsou opět průměry přes 10 evaluací na testovacích částech datových sad.

Ze srovnání vychází nejhůře model ALBERT. To není nijak překvapivé, jelikož hlavní výhodou této architektury je trénování výrazně větších modelů na úkor horších výsledků při

srovnání s ostatními architekturami. Modely založené na předtrénované síti $\text{ALBERT}_{\text{xxlarge}}$ dosahují jedněch z nejlepších výsledků napříč různými úlohami, ale pro praktické užití implementovaného systému se musel udělat kompromis mezi úspěšností klasifikace a nároky modelu na prostředky.

Při porovnání modelů BERT a XLNet, dosahuje na datové sadě *SemEval* model BERT lepších výsledků, které jsou podobné jako u modelu R-BERT (vizte tabulku 5.1), ale na ostatních datových sadách ho s výraznějším odstupem překonává model XLNet. Datové sady *KBP37* a *RCD-FewRel* obsahují větší počet konkrétnějších vztahů, delší věty a víceslovné pojmenované entity, které se nacházejí od sebe ve větším rozestupu. Tyto vztahy by tedy šlo označit za složitější ke klasifikaci, než vztahy v datové sadě *SemEval*.

Posledním srovnávaným modelem je model RoBERTa. Ten sice dosahuje na datových sadách *SemEval* a *RCD-FewRel* horších výsledků než modely BERT a XLNet, ale překvapivě je výrazně překonává na datové sadě *KBP37*. Případy v této datové sadě se hlavně vyznačují větším počtem slov. Z toho lze usuzovat, že model RoBERTa dokáže vytvářet lepší kontextové reprezentace pro dlouhé věty, vhodné pro klasifikaci vztahů mezi pojmenovanými entitami.

5.4 Shrnutí

V úvodu kapitoly jsou představeny použité metriky a datové sady. Na základě *FewRel* byla vytvořena nová datová sada *RDC-FewRel*, která je rozšířena o negativní případy. Datové sady byly v kapitole analyzovány a srovnány. Experimenty se zaměřily na hledání hyperparametrů, způsob reprezentace víceslovných entit, reprezentace dvojice entit a různých předtrénovaných modelů sítí typu *transformers*. Výsledky ukázaly, že na rozdíl od modelu RoBERTa nejsou modely BERT a XLNet tolik úspěšné při klasifikaci vztahů v dlouhých větách. Na datových sadách s průměrně kratšími větami naopak dosahují lepších výsledků.

Kapitola 6

Závěr

V práci byl navržen systém pro extrakci vztahů mezi pojmenovanými entitami ze stažených souborů na webu. Systém je modulární, kdy je možné nastavit celý proces zpracování, od vybrání formátu vstupních dat, přes model pro získávání pojmenovaných entit a model pro klasifikaci vztahů, až po formát výstupního souboru. Systém je navržen tak, aby šel jednoduše rozšířit o další moduly nebo podporované formáty. Ve spojení s inkrementálními webovými prohledávacími moduly (angl. *web crawler*) lze systém využít k automatizovanému vytváření znalostních bází.

Model pro klasifikaci vztahů je postaven na předtrénovaném modelu sítě typu *transformers*, které dosahují v oblasti zpracování přirozeného jazyka velmi dobrých výsledků. Práce také obsahuje krátkou rešerši předtrénovaných sítí BERT, XLNet, RoBERTa a ALBERT, se kterými bylo následně experimentováno.

Se systémem bylo provedeno několik experimentů na třech datových sadách. Pro účely této práce byla vytvořena datová sada *RCD-FewRel*, která je založena na datové sadě *FewRel* a obsahuje 80 vztahů ve 63725 větách, včetně negativních případů. Samotné experimenty se zaměřily na využití různých předtrénovaných modelů a nuance architektury. Model pro klasifikaci vztahů se umístil mezi nejlepšími řešeními na datové sadě *SemEval 2010 Task 8*, která je hojně využívána k porovnání výsledků.

Budoucí práce můžou navázat přidáním další analýzy přirozeného jazyka do systému nebo podporou jiných vstupních formátů, jako např. formát PDF. Při extrakci vztahů lze sloučit rozpoznávání pojmenovaných entit s modelem pro klasifikaci vztahů, natrénovat vlastní síť typu *transformers*, nebo experimentovat s méně náročnými modely na zdroje. Vytvořená datová sada *RCD-FewRel* může být rozšířena o další vztahy a vzorky z různých domén, nebo navržením komplexnější metody pro získání negativních případů, případně jejich manuální anotací.

Literatura

[1] Akbik, A., Blythe, D. a Vollgraf, R. Contextual String Embeddings for Sequence Labeling. In: Bender, E. M., Derczynski, L. a Isabelle, P., ed. *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, 2018, s. 1638–1649. ISBN 978-1-948087-50-6. Dostupné z: https://www.aclweb.org/anthology/C18-1139.

[2] Alt, C., Hübner, M. a Hennig, L. Improving Relation Extraction by Pre-trained Language Representations. *ArXiv*. 2019, abs/1906.03088. Dostupné z: https://arxiv.org/abs/1906.03088.

[3] Bach, N. a Badaskar, S. A Review of Relation Extraction. *Literature review for Language and Statistics II*. 2007. Dostupné z: https://www.cs.cmu.edu/~nbach/papers/A-survey-on-Relation-Extraction.pdf.

[4] Baevski, A., Edunov, S., Liu, Y., Zettlemoyer, L. a Auli, M. Cloze-driven Pretraining of Self-attention Networks. In: Padó, S. a Huang, R., ed. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, 2019, s. 5359–5368. ISBN 978-1-950737-90-1. Dostupné z: https://www.aclweb.org/anthology/D19-1539.

[5] Baldini Soares, L., FitzGerald, N., Ling, J. a Kwiatkowski, T. Matching the Blanks: Distributional Similarity for Relation Learning. In: Korhonen, A., Traum, D. a Màrquez, L., ed. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Florence, Italy: Association for Computational Linguistics, 2019, s. 2895–2905. ISBN 978-1-950737-48-2. Dostupné z: https://www.aclweb.org/anthology/P19-1279.

[6] Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M. a Etzioni, O. Open Information Extraction from the Web. In: Veloso, M. M., ed. *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*. Hyderabad, India: ijcai.org, Leden 2007, s. 2670–2676. Dostupné z: http://ijcai.org/Proceedings/07/Papers/429.pdf.

[7] Bast, H. a Buchhold, B. QLever: A Query Engine for Efficient SPARQL+Text Search. In: Lim, E.-P. a Winslett, M., ed. *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*. New York, NY, USA: Association for Computing Machinery, 2017, s. 647–656. ISBN 978-1-450349-18-5. Dostupné z: https://doi.org/10.1145/3132847.3132921.

[8] Boldi, P., Marino, A., Santini, M. a Vigna, S. BUbiNG: Massive Crawling for the Masses. *ACM Trans. Web*. New York, NY, USA: Association for Computing Machinery. červen 2018, sv. 12, č. 2. ISSN 1559-1131. Dostupné z: https://doi.org/10.1145/3160017.

[9] Boldi, P. a Vigna, S. MG4J at TREC 2005. In: Voorhees, E. M. a Buckland, L. P., ed. *Proceedings of the Fourteenth Text REtrieval Conference (TREC 2005)*. NIST, 2005, SP 500-266, s. 4. Special Papers. Dostupné z: http://mg4j.di.unimi.it/.

[10] Choi, J. D., Tetreault, J. a Stent, A. It Depends: Dependency Parser Comparison Using A Web-based Evaluation Tool. In: Zong, C. a Strube, M., ed. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, 2015, s. 387–396. ISBN 978-1-941643-72-3. Dostupné z: https://www.aclweb.org/anthology/P15-1038.

[11] Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. et al. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In: Korhonen, A., Traum, D. a Màrquez, L., ed. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, červenec 2019, s. 2978–2988. ISBN 978-1-950737-48-2. Dostupné z: https://www.aclweb.org/anthology/P19-1285.

[12] Devlin, J., Chang, M.-W., Lee, K. a Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: Burstein, J., Doran, C. a Solorio, T., ed. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019, s. 4171–4186. ISBN 978-1-950737-13-0. Dostupné z: https://www.aclweb.org/anthology/N19-1423.

[13] Finkel, J. R., Grenager, T. a Manning, C. Incorporating non-local information into information extraction systems by gibbs sampling. In: Knight, K., Ng, H. T. a Oflazer, K., ed. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. University of Michigan, USA: Association for Computational Linguistics, 2005, s. 363–370. Dostupné z: https://www.aclweb.org/anthology/P05-1045.

[14] Han, X., Zhu, H., Yu, P., Wang, Z., Yao, Y. et al. FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation. In: Riloff, E., Chiang, D., Hockenmaier, J. a Tsujii, J., ed. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, 2018, s. 4803–4809. ISBN 978-1-948087-84-1. Dostupné z: https://www.aclweb.org/anthology/D18-1514.

[15] Hendrickx, I., Kim, S. N., Kozareva, Z., Nakov, P., Ó Séaghdha, D. et al. SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals. In: Erk, K. a Strapparava, C., ed. *Proceedings of the 5th International Workshop on Semantic Evaluation*. Uppsala, Sweden: Association for

Computational Linguistics, 2010, s. 33–38. ISBN 978-1-932432-70-1. Dostupné z: https://www.aclweb.org/anthology/S10-1006.

[16] Hochreiter, S. a Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* Cambridge, MA, USA: MIT Press. 1997, sv. 9, č. 8, s. 1735–1780. ISSN 0899-7667. Dostupné z: https://doi.org/10.1162/neco.1997.9.8.1735.

[17] Jiang, Y., Hu, C., Xiao, T., Zhang, C. a Zhu, J. Improved Differentiable Architecture Search for Language Modeling and Named Entity Recognition. In: Inui, K., Jiang, J., Ng, V. a Wan, X., ed. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, 2019, s. 3585–3590. ISBN 978-1-950737-90-1. Dostupné z: https://www.aclweb.org/anthology/D19-1367.

[18] Kim, Y. Convolutional Neural Networks for Sentence Classification. In: Moschitti, A., Pang, B. a Daelemans, W., ed. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014, s. 1746–1751. ISBN 978-1-937284-96-1. Dostupné z: https://www.aclweb.org/anthology/D14-1181.

[19] Kim, Y., Jernite, Y., Sontag, D. a Rush, A. M. Character-Aware Neural Language Models. In: Schuurmans, D. a Wellman, M., ed. *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence and the Twenty-Eighth Innovative Applications of Artificial Intelligence Conference Volume One*. AAAI Press, 2016, s. 2741–2749. AAAI'16. ISBN 978-1-577357-61-2. Dostupné z: https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/viewPaper/12489.

[20] Konkol, M. a Konopík, M. Segment Representations in Named Entity Recognition. In: Král, P. a Matoušek, V., ed. *Text, Speech, and Dialogue*. Springer International Publishing, 2015, sv. 9302, s. 61–70. Lecture Notes in Computer Science. ISBN 978-3-319-24032-9. Dostupné z: http://dx.doi.org/10.1007/978-3-319-24033-6_7.

[21] Lafferty, J. D., McCallum, A. a Pereira, F. C. N. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: Brodley, C. E. a Danyluk, A. P., ed. *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, s. 282–289. ISBN 978-1-55860-778-1. Dostupné z: https://repository.upenn.edu/cgi/viewcontent.cgi?article=1162&context=cis_papers.

[22] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K. a Dyer, C. Neural Architectures for Named Entity Recognition. In: Knight, K., Nenkova, A. a Rambow, O., ed. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, 2016, s. 260–270. ISBN 978-1-941643-91-4. Dostupné z: https://www.aclweb.org/anthology/N16-1030.

[23] Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P. et al. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In: Song, D., Cho, K. a White, M., ed. *International Conference on Learning Representations*. Addis

Ababa, Ethiopia: OpenReview.net, Duben 2020. Dostupné z: https://openreview.net/forum?id=H1eA7AEtvS.

[24] Lee, J., Seo, S. a Choi, Y. S. Semantic Relation Classification via Bidirectional LSTM Networks with Entity-Aware Attention Using Latent Entity Typing. *Symmetry*. MDPI AG. Jun 2019, sv. 11, č. 6, s. 785. ISSN 2073-8994. Dostupné z: https://arxiv.org/abs/1901.08163.

[25] Levy, O., Seo, M., Choi, E. a Zettlemoyer, L. Zero-Shot Relation Extraction via Reading Comprehension. In: Levy, R. a Specia, L., ed. *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. Vancouver, Canada: Association for Computational Linguistics, 2017, s. 333–342. ISBN 978-1-945626-54-8. Dostupné z: https://www.aclweb.org/anthology/K17-1034.

[26] Li, C. a Tian, Y. Downstream Model Design of Pre-trained Language Model for Relation Extraction Task. *ArXiv*. 2020, abs/2004.03786. Dostupné z: https://arxiv.org/abs/2004.03786.

[27] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M. et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv*. 2019, abs/1907.11692. Dostupné z: https://arxiv.org/abs/1907.11692.

[28] McCallum, A., Freitag, D. a Pereira, F. C. N. Maximum Entropy Markov Models for Information Extraction and Segmentation. In: Langley, P., ed. *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, s. 591–598. ISBN 978-1-55860-707-2. Dostupné z: http://www.ai.mit.edu/courses/6.891-nlp/READINGS/maxent.pdf.

[29] Mikolov, T., Chen, K., Corrado, G. a Dean, J. Efficient Estimation of Word Representations in Vector Space. *ICLR Workshop*. 2013. Dostupné z: https://arxiv.org/abs/1301.3781.

[30] Mikolov, T., Yih, W.-t. a Zweig, G. Linguistic Regularities in Continuous Space Word Representations. In: Vanderwende, L., Daumé III, H. a Kirchhoff, K., ed. *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia: Association for Computational Linguistics, 2013, s. 746–751. ISBN 978-1-937284-47-3. Dostupné z: https://www.aclweb.org/anthology/N13-1090.

[31] Ondřej, K. *Inkrementální stahování webu pomocí systému Bubing*. Brno, CZ, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Dostupné z: https://www.fit.vut.cz/study/thesis/20994/.

[32] Pennington, J., Socher, R. a Manning, C. Glove: Global Vectors for Word Representation. In: Moschitti, A., Pang, B. a Daelemans, W., ed. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014, s. 1532–1543. ISBN 978-1-937284-96-1. Dostupné z: https://www.aclweb.org/anthology/D14-1162.

[33] Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C. et al. Deep Contextualized Word Representations. In: Walker, M., Ji, H. a Stent, A., ed. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018, s. 2227–2237. ISBN 978-1-948087-27-8. Dostupné z: https://www.aclweb.org/anthology/N18-1202.

[34] Peters, M. E., Neumann, M., Logan, R., Schwartz, R., Joshi, V. et al. Knowledge Enhanced Contextual Word Representations. In: Inui, K., Jiang, J., Ng, V. a Wan, X., ed. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Listopad 2019, s. 43–54. ISBN 978-1-950737-90-1. Dostupné z: https://www.aclweb.org/anthology/D19-1005.

[35] Radford, A., Narasimhan, K., Salimans, T. a Sutskever, I. *Improving Language Understanding by Generative Pre-Training*. 2018. Dostupné z: https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf.

[36] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. et al. Language models are unsupervised multitask learners. *OpenAI Blog*. 2019, sv. 1, č. 8. Dostupné z: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

[37] Ratinov, L. a Roth, D. Design Challenges and Misconceptions in Named Entity Recognition. In: Stevenson, S. a Carreras, X., ed. *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*. Boulder, Colorado: Association for Computational Linguistics, 2009, s. 147–155. ISBN 978-1-932432-29-9. Dostupné z: https://www.aclweb.org/anthology/W09-1119.

[38] Shen, Y. a Huang, X. Attention-Based Convolutional Neural Network for Semantic Relation Extraction. In: Matsumoto, Y. a Prasad, R., ed. *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan: The COLING 2016 Organizing Committee, 2016, s. 2526–2536. ISBN 978-4-87974-702-0. Dostupné z: https://www.aclweb.org/anthology/C16-1238.

[39] Staudemeyer, R. C. a Morris, E. R. Understanding LSTM – a tutorial into Long Short-Term Memory Recurrent Neural Networks. *ArXiv*. 2019, abs/1909.09586. Dostupné z: https://arxiv.org/abs/1909.09586.

[40] Straková, J., Straka, M. a Hajic, J. Neural Architectures for Nested NER through Linearization. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, 2019, s. 5326–5331. ISBN 978-1-950737-48-2. Dostupné z: https://www.aclweb.org/anthology/P19-1527.

[41] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L. et al. Attention is All You Need. In: Luxburg, U. von, Guyon, I., Bengio, S., Wallach, H.

a Fergus, R., ed. *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2017, s. 6000–6010. ISBN 978-1-5108-6096-4. Dostupné z: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

[42] Wang, H., Tan, M., Yu, M., Chang, S., Wang, D. et al. Extracting Multiple-Relations in One-Pass with Pre-Trained Transformers. In: Korhonen, A., Traum, D. a Màrquez, L., ed. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, červenec 2019, s. 1371–1377. ISBN 978-1-950737-48-2. Dostupné z: https://www.aclweb.org/anthology/P19-1132.

[43] Wang, L., Cao, Z., Melo, G. de a Liu, Z. Relation Classification via Multi-Level Attention CNNs. In: Erk, K. a Smith, N. A., ed. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, 2016, s. 1298–1307. ISBN 978-1-945626-00-5. Dostupné z: https://www.aclweb.org/anthology/P16-1123.

[44] Wu, S. a He, Y. Enriching Pre-trained Language Model with Entity Information for Relation Classification. In: Cui, P., Rundensteiner, E., Carmel, D., He, Q. a Yu, J. X., ed. *Proceedings of the 28th ACM International Conference on Information and Knowledge Managements*. ACM 2019, 2019, s. 2361–2364. ISBN 978-1-4503-6976-3. Dostupné z: http://dx.doi.org/10.1145/3357384.3358119.

[45] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R. et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In: Wallach, H., Larochelle, H., Beygelzimer, A., Alché Buc, F. d', Fox, E. et al., ed. *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, s. 5753–5763. Dostupné z: https://papers.nips.cc/paper/8812-xlnet-generalized-autoregressive-pretraining-for-language-understanding.

[46] Zeng, D., Liu, K., Lai, S., Zhou, G. a Zhao, J. Relation Classification via Convolutional Deep Neural Network. In: Tsujii, J. a Hajic, J., ed. *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Dublin, Ireland: Dublin City University and Association for Computational Linguistics, 2014, s. 2335–2344. ISBN 978-1-941643-26-6. Dostupné z: https://www.aclweb.org/anthology/C14-1220.

[47] Zhang, D. a Wang, D. Relation Classification via Recurrent Neural Network. *ArXiv*. 2015, abs/1508.01006. Dostupné z: https://arxiv.org/abs/1508.01006.

[48] Zhang, S., Zheng, D., Hu, X. a Yang, M. Bidirectional Long Short-Term Memory Networks for Relation Classification. In: Zhao, H., ed. *Proceesings of the 29th Pacific Asia Conference on Language, Information and Computation (PACLIC 2015)*. Shanghai, China: ACL 2015, 2015, s. 73–78. ISBN 978-1-5108-1756-2. Dostupné z: https://www.aclweb.org/anthology/Y15-1009.

[49] Zhao, Y., Wan, H., Gao, J. a Lin, Y. Improving Relation Classification by Entity Pair Graph. In: Lee, W. S. a Suzuki, T., ed. *Proceedings of The Eleventh Asian Conference on Machine Learning*. Nagoya, Japan: PMLR, Nov 2019, sv. 101, s. 1156–1171. Proceedings of Machine Learning Research. Dostupné z: http://proceedings.mlr.press/v101/zhao19a.html.

[50] Zhou, P., Shi, W., Tian, J., Qi, Z., Li, B. et al. Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. In: Erk, K. a Smith, N. A., ed. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Berlin, Germany: Association for Computational Linguistics, 2016, s. 207–212. ISBN 978-1-945626-01-2. Dostupné z: https://www.aclweb.org/anthology/P16-2034.

[51] Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R. et al. Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books. In: O'Conner, L., ed. *2015 IEEE International Conference on Computer Vision (ICCV)*. Santiago, Chile: IEEE Computer Society 2015, 2015, s. 19–27. ISBN 978-1-4673-8390-5. Dostupné z: http://ieeexplore.ieee.org/document/7410368/.

Příloha A

Obsah přiloženého paměťového média

- `README.md` – Pokyny k datovýcm sadám a aplikaci.
- `dip/` – Text této práce.
- `src/` – Zdrojové soubory.
  - `re/` – Repozitář extrakčního systému.
  - `re/lib/remodel/` – Repozitář modelu pro klasifikaci.
  - `scripts/` – Všechny pomocné skripty.
- `data/` – Výsledky experimentů a vytvořená datová sada.
- `xondre09-dip-poster.pdf` – Plakát reprezentující práci, její cíle a výsledky.

Příloha B

Datové sady

B.1 KBP37

Vztahy mezi pojmenovanými entitami	
per:alternate_names	org:alternate_names
per:cities_of_residence	org:city_of_headquarters
per:countries_of_residence	org:country_of_headquarters
per:country_of_birth	org:founded_by
per:employee_of	org:founded
per:origin	org:members
per:spouse	org:stateorprovince_of_headquarters
per:stateorprovinces_of_residence	org:subsidiaries
per:title	org:top_members/employees
no_relation	

Tabulka B.1: Vztahy mezi pojmenovanými entitami v datové sadě *KBP37*.

B.2 FewRel

Kód vztahu	Vztah slovně	Tři nejčastější slova	Nejčastější dvojice slov
P6	head	minister mayor prime	minister–prime states–united president–minister
P17	country	united states river	states–united zealand–new minister–prime
P22	father	son daughter father	son–eldest sons–two prince–crown

Pokračování na další straně...

Tabulka B.2: Analýza datové sady *FewRel*.

Kód vztahu	Vztah slovně	Tři nejčastější slova	Nejčastější dvojice slov
P25	mother	daughter son mother	wife–first wife–second sons–two
P26	spouse	wife daughter son	wife–first wife–second son–eldest
P27	country	born american united	states–united olympics–summer summer–competed
P31	instance	film game video	game–video system–operating parish–civil
P39	position	minister president governor	minister–prime states–united general–attorney
P40	child	son daughter father	son–eldest king–son children–two
P57	director	film directed director	directed–film film–drama directed–written
P58	screenwriter	film written directed	directed–film wrote–co novel–based
P59	constellation	constellation ngc galaxy	galaxy–spiral years–light constellation–located
P84	architect	designed architect building	architect–designed designed–also designed–building
P86	composer	music composed film	composed–music score–film premiere–world
P101	field	university born theory	theory–number science–computer laureate–nobel
P102	member	party republican election	party–labour party–democratic election–general

Pokračování na další straně...

Tabulka B.2: Analýza datové sady *FewRel*.

Kód vztahu	Vztah slovně	Tři nejčastější slova	Nejčastější dvojice slov
P105	taxon	family species genus	gastropod–marine sea–species mollusk–gastropod
P106	occupation	born american film	director–film cyclist–racing screenwriter–director
P118	league	league born season	league–football plays–footballer league–premier
P123	publisher	published game released	game–video developed–game press–university
P127	owned	company group owned	york–new station–railway trust–national
P131	located	county district located	states–united york–new park–national
P135	movement	baroque art painter	renaissance–harlem art–pop baroque–early
P136	genre	band album rock	band–rock band–metal album–studio
P137	operator	line station class	states–united force–air station–railway
P140	religion	catholic church jewish	church–catholic catholic–roman priest–catholic
P150	contains	county district province	states–united community– unincorporated district–administrative
P155	follows	elected album election	olympics–summer election–general elections–general

Pokračování na další straně...

Tabulka B.2: Analýza datové sady *FewRel*.

Kód vztahu	Vztah slovně	Tři nejčastější slova	Nejčastější dvojice slov
P156	followed	elected album released	olympics–summer election–general championships–world
P159	headquarters	based new company	york–new city–york states–united
P175	performer	album song released	album–studio band–rock album–debut
P176	manufacturer	built company class	motors–general company–motor states–united
P177	crosses	bridge river across	across–bridge crosses–bridge river–mississippi
P178	developer	game developed games	game–video developed–game playing–role
P206	located	island river sea	ocean–pacific thames–river ocean–atlantic
P241	military	army states british	states–united army–british navy–states
P264	record	records released album	album–studio album–debut american–album
P276	location	held world new	york–new prix–grand championship–open
P306	operating	windows android microsoft	windows–microsoft x–os os–mac
P355	subsidiary	university group national	states–united university–state department–states
P361	part	season episode station	olympics–summer series–television war–world

Pokračování na další straně...

Tabulka B.2: Analýza datové sady *FewRel*.

Kód vztahu	Vztah slovně	Tři nejčastější slova	Nejčastější dvojice slov
P364	original	film directed hindi	directed–film film–hindi film–malayalam
P400	platform	game released playstation	windows–microsoft game–video xbox–playstation
P403	mouth	river tributary flows	tributary–river romania–river flows–river
P407	language	language english french	language–english language–french name–given
P410	military	general army officer	army–british general–major officer–army
P412	voice	soprano born operatic	soprano–operatic soprano–mezzo operatic–american
P413	position	born footballer plays	plays–footballer footballer–professional played–footballer
P449	original	series television show	series–television one–bbc series–drama
P460	said	name given form	name–given include–name form–diminutive
P463	member	member band members	nations–united society–royal sciences–academy
P466	occupant	stadium home played	games–home ground–home home–play
P495	country	united states american	states–united band–rock album–studio
P527	has	season band episode	series–television american–season television–animated

Pokračování na další straně...

Tabulka B.2: Analýza datové sady *FewRel*.

Kód vztahu	Vztah slovně	Tři nejčastější slova	Nejčastější dvojice slov
P551	residence	born new city	york–new city–york states–united
P641	sport	born player football	olympics–summer hockey–ice football–american
P674	characters	character film series	character–fictional turtles–ninja ninja–mutant
P706	located	island dam river	ridge–metacomet islands–shetland south–island
P710	participant	contest song eurovision	song–eurovision contest–song contest–representative
P740	location	band album based	band–rock york–new band–metal
P750	distributor	film pictures released	pictures–paramount pictures–columbia pictures–universal
P800	notable	novel also first	name–novel novel–based adaptation–film
P921	main	war world film	war–world ii–war world–second
P931	place	airport international located	airport–international located–airport airport–airport
P937	work	london born new	york–new city–york artists–union
P974	tributary	river tributary flows	tributary–river romania–river flows–river
P991	successful	election elected party	election–general election–presidential minister–prime

Pokračování na další straně...

Tabulka B.2: Analýza datové sady *FewRel*.

Kód vztahu	Vztah slovně	Tři nejčastější slova	Nejčastější dvojice slov
P1001	applies	state governor states	states–united general–attorney union–european
P1303	instrument	player bass guitar	guitar–bass player–bass bass–double
P1344	participant	olympics summer competed	olympics–summer summer–competed olympics–winter
P1346	winner	first award cup	best–award prix–grand prize–peace
P1408	licensed	radio station fm	station–radio licensed–station york–new
P1411	nominated	best academy award	award–academy best–award academy–nominated
P1435	heritage	historic national register	places–historic historic–register register–national
P1877	after	novel film based	novel–based adaptation–film name–novel
P1923	participating	cup world fifa	cup–world world–fifa team–national
P2094	competition	title champion boxer	boxer–professional title–middleweight heavyweight–light
P3373	sibling	brother brothers sister	brother–younger sons–three brother–elder
P3450	sports	season league cup	league–football cup–world championships–world
P4552	mountain	mountains range mount	land–victoria antarctica–mountains maud–queen

Tabulka B.2: Analýza datové sady *FewRel*.

Příloha C

Náhled plakátu